UNIVERSIDAD CARLOS III DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

TRABAJO FIN DE GRADO

DISEÑO DE ALGORITMO DE DETECCIÓN DE OBSTÁCULOS PARA VEHÍCULO INTELIGENTE BASADO EN VISIÓN POR COMPUTADOR

Autor: Alberto Pastor Soriano
Tutor: Aurelio Ponz Vila

Índice

Índice de figuras

Índice de tablas

1. Introducción

1.1. Motivación.

Los sistemas de ayuda a la conducción están presentes en la sociedad actual. El desarrollo de la tecnología en las últimas décadas ha permitido al ser humano introducir mejoras en los vehículos.

Hoy en día, la mayor parte de los accidentes en carretera se producen por errores del conductor. Los ADAS (Sistemas Avanzados de Ayuda a la Conducción) buscan reducir el error humano, tratando de detectar y anticipar las situaciones peligrosas, avisando al conductor para que no se produzca el accidente.

La evolución de los fallecidos en accidente de tráfico se observan en la siguiente figura:

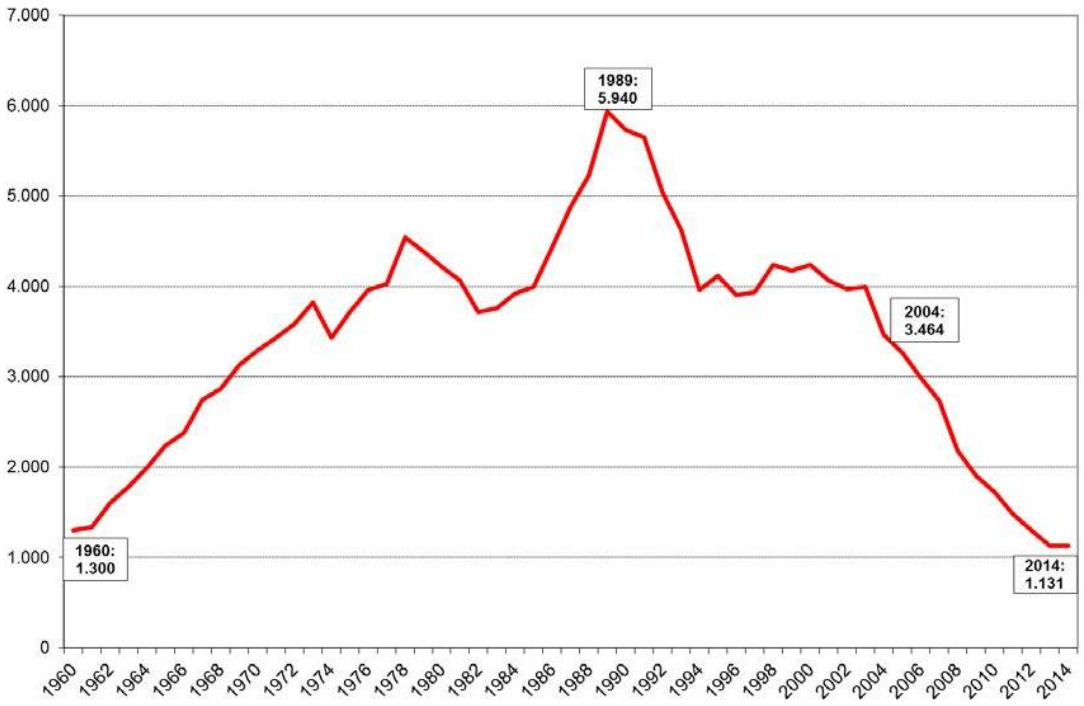


Figura 1: Evolución de fallecidos en accidente en los últimos años

Como se puede observar el número de víctimas en accidentes entre 2004 y 2014 ha disminuido aproximadamente un 64%.

Según fuentes de la Dirección General de Tráfico (DGT) [1] si analizamos el número de accidentes según el tipo de vía, podemos decir que es en las vías urbanas, concretamente en las calles, donde se producen la gran mayoría de los accidentes. No obstante, el mayor número de fallecidos es en las carreteras convencionales (véase *Figura 2*).

Figura 2: Disminución del número de accidentes según su localización.

El principal motivo de que en las carreteras convencionales se produzca el mayor número de fallecidos, es debido a que este tipo de vía es de doble sentido, aumentando así la posibilidad de que se produzcan tanto colisiones con otros vehículos como salidas de vía o incluso atropellos a peatones.

La motocicleta es positiva para las ciudades porque descongestiona, ocupa menos espacio, reduce los tiempos en los traslados y también las emisiones nocivas. La principal problemática es que, aunque se produzcan menos accidentes que en relación a los coches y camionetas, si los relacionamos con las muertes y lesiones, las motocicletas ocupan el primer lugar (véase *Figura 3*). En Madrid el 39% de las víctimas graves en accidentes de tráfico son motoristas.

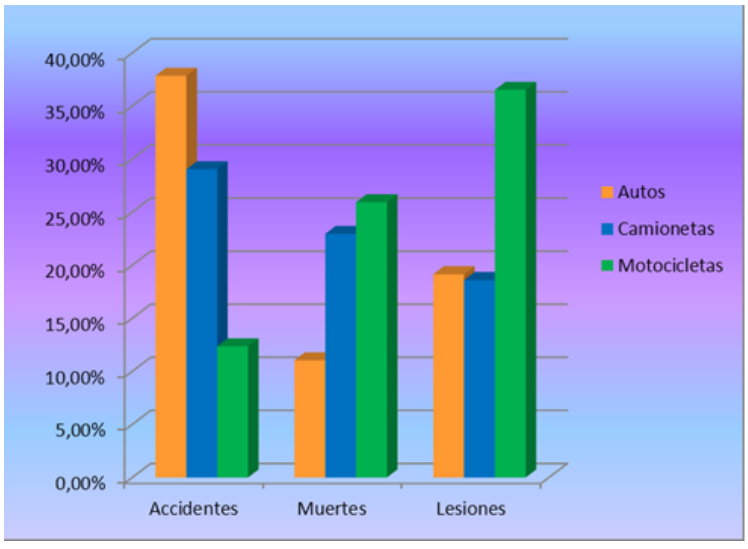


Figura 3: Accidentes, muertes y lesiones según vehículo

Los fabricantes de tecnologías trabajan una futura mejora que prevea mayor protección contra accidentes. Estos desarrollos dirigidos a una conducción automática, tiene el objetivo de conseguir una conducción libre de accidentes y sin lesiones.

En Europa, además, el sistema de valoración de EuroNCAP es un importante motor para la divulgación de las funciones que mejoran la seguridad. En un principio, esta seguridad era pasiva, es decir, se encargaba de minimizar los posibles daños en caso de que llegase a producir un accidente. Posteriormente, y gracias al avance tecnológico, se han desarrollado un conjunto de mecanismos o dispositivos destinados a disminuir el riesgo de que se produzca un accidente (seguridad activa).

1.2. Objetivo del proyecto.

El objetivo principal de este proyecto es el desarrollo de un algoritmo basado en la visión por computador para la detección de motocicletas en entorno viario para la posible implantación en el vehículo inteligente de la Universidad Carlos III de Madrid.

En primer lugar se pretende generar un conjunto de entrenamiento bastante amplio y con varios modelos de motocicletas y vehículos.

Se explicarán los conceptos básicos del funcionamiento de una cámara estéreo, la cual ha sido utilizada para obtener las secuencias para el entrenamiento del clasificador.

Por otro lado, se pretende conseguir un algoritmo de detección rápido, robusto y eficiente que sea capaz de cumplir unos requisitos mínimos computación. Además al presente proyecto se le puede añadir un sistema de alerta y la realización de una frenada de emergencia.

El algoritmo ha sido desarrollado en los lenguajes de programación C/C++, basándose en las librerías OpenCV y Boost, en el entorno de desarrollo integrado Qt Creator y en el sistema operativo Linux.

2. Estado del arte

2.1. Introducción.

El Estado del Arte del presente proyecto se centra en la evolución de los sistemas de seguridad para los vehículos así como su funcionalidad de aquellos se encuentran tanto en desarrollo como en el mercado. Posteriormente se explicarán tanto el programa europeo de evaluación EuroNCAP y el concepto de fusión sensorial. Por ende se hará referencia a los vehículos autónomos.

2.2. Evolución de los sistemas de seguridad para los automóviles.

Durante los últimos años, el uso del automóvil como medio de transporte se ha incrementado notablemente, como consecuencia de esto y aunque se hayan tomado grandes medidas para disminuir el número de víctimas de accidentes de tráfico, éstas siguen siendo muy elevadas.

La seguridad en los vehículos ha aumentado mucho en los últimos años gracias a las innovaciones tecnológicas. Estas innovaciones van desde los sensores que detectan el peligro de colisión, sistemas pre-safe, hasta los sistemas que permiten facilitan la conducción al usuario.

En un principio los desarrollos se centraban en minimizar los posibles daños de los ocupantes del vehículo en caso de que se produjera un accidente, es decir, en la seguridad pasiva. En esta seguridad se engloban desde el diseño de estructuras de deformación del automóvil para la absorción de la energía en caso de impacto hasta los principales accesorios de seguridad como los cinturones o los airbag.

Gracias a la introducción de la informática y sobretodo de la electrónica, tanto analógica como digital, empiezan a surgir en la década de los 70 una serie de sistemas orientados a la seguridad activa, es decir, mecanismos o dispositivos destinados a disminuir el riesgo de que se produzca un accidente. Un ejemplo son los siguientes sistemas: el ABS (Bosch 1970) cuya función es evitar que se bloqueen las ruedas al producirse un frenazo intenso, modulando automáticamente la presión sobre los frenos cuando el conductor pisa el pedal a fondo. El ESP (Bosch 1995) son las iniciales en alemán del “control electrónico de estabilidad” que detecta la desviación de la trayectoria del vehículo con respecto a la dirección que se desea llevar, el sistema frena ligeramente cada rueda por separado y así devuelve el vehículo a la trayectoria deseada.

Si todos los vehículos incorporaran ESP y ABS de serie se podría contribuir a reducir un 30% los accidentes relacionados con frenadas de emergencia y evasión de objetos en la trayectoria del vehículo.

Son tantos los beneficios de este sistema en la reducción de siniestros, que a partir de noviembre de 2011, los vehículos de turismo e industriales ligeros ya incorporan como elemento de serie el ESP para los estados de la Unión Europea. Este dato nos puede dar una idea de la importancia de los sistemas tecnológicos incorporados en los automóviles.

Posteriormente, como consecuencia de los anteriores dispositivos y con una evolución notable de la electrónica y de la comunicación entre los sensores que abordan en un vehículo, surgen los sistemas ADAS que sirven de gran ayuda para ayudar al conductor para evitar accidentes. Estos sistemas intentan de manera individual disminuir los tipos de accidentes mortales y no mortales más frecuentes, avisando al conductor o tomando medidas preventivas siendo, según las estadísticas de la DGT para el año 2009, la salida de la vía fue el más frecuente, seguido de colisiones frontal y fronto-lateral y atropellos a peatones, por este orden.

Esto significa que un sistema como el diseñado en este proyecto para la detección y reconocimiento de motocicletas o cualquier otro obstáculo, se podría utilizar para evitar el segundo tipo de accidente más habitual.

2.3. Sistemas ADAS (Advanced Driver Assistance Systems).

La asistencia al conductor busca incrementar la seguridad global y aumentar las capacidades de respuesta en la conducción. Estos sistemas tratan de desarrollar tecnologías inteligentes aplicadas al automóvil con el propósito tanto de aumentar la seguridad en las carreteras consiguiendo reducir el número de accidentes, como de facilitar la conducción para que ésta sea más confortable. Los ADAS pueden cubrir un completo rango de sistemas que varían desde sistemas que proporcionan información o avisos, hasta sistemas que intervienen en el control y en las tareas de maniobra del vehículo.

La implementación de estos sistemas en los vehículos es reciente. Se podría considerar que el primer sistema ADAS integrado en un vehículo de serie fue el “Adaptative Cruise Control” en 1995, y que fue incorporado en un vehículo para el mercado japonés. No sería sino en 1998 cuando Mercedes- Benz introdujo por primera vez este sistema en Europa. Este dispositivo consiste básicamente en controlar la velocidad del vehículo según la distancia de seguridad con el vehículo precedente.

Estos primeros dispositivos se empezaron a instalar solamente en coches de gama alta y a precios elevados, pero en la actualidad cada vez es más común que se integren en vehículos de clase media, y se espera que en los próximos años la instalación tenga una crecida casi exponencial, ayudando a reducir los accidentes de tráfico y las consecuencias que estos provocan sobre las personas.

En la *Figura 4*, se muestran algunos ejemplos de sistemas ADAS como son el ACC (Adaptative Cruise Control), el AFS (Adaptative Frontlighting System), LDW

(Lane Departure Warning), LCA (Lane Change Assistant). TRS (Traffic Sign Recognition) y Sistema anticolisión entre otros.

Figura 4: Sistemas ADAS

Entre los ADAS se pueden distinguir sistemas destinados a dar soporte a varios aspectos de la conducción proporcionando información, comúnmente conocidos como IVIS (In-Vehicle Information Systems). Por otro lado, se encuentran los sistemas de alerta o reacción, cuyo objetivo es reducir los errores humanos. Los medios de alertar al conductor pueden ser auditivos, visuales o por contacto, como por ejemplo mediante la vibración del asiento o del volante. Otro tipo de ADAS son aquellos que intervienen en el control del vehículo pero sin suplantar completamente al conductor, como lo harían los vehículos denominados “vehículos autónomos”, en el que el conductor deja de controlar el vehículo en su totalidad.

A continuación se describirán los sistemas de asistencia a la conducción más relevantes:

2.3.1. Control de crucero adaptativo (ACC).

El Control de Crucero Adaptativo (Adaptative Cruise Control, ACC), también denominado control de velocidad inteligente, aumenta los beneficios del control de crucero, que permite establecer una velocidad de conducción y mantenerla automáticamente sin necesidad de pisar el acelerador, con una dimensión adicional: la regulación automática de la distancia de seguridad con el vehículo precedente. Si el vehículo que va delante reduce la velocidad o se detecta otro objeto, el ACC ajusta la velocidad del vehículo sin intervención del conductor. Cuando se haya despejado la carretera, el sistema vuelve a acelerar el vehículo hasta la velocidad programada y avisa al conductor, generalmente por medio de una señal acústica. Los actuales sistemas ACC son ante todo dispositivos de confort previstos para una gama de velocidades limitada.

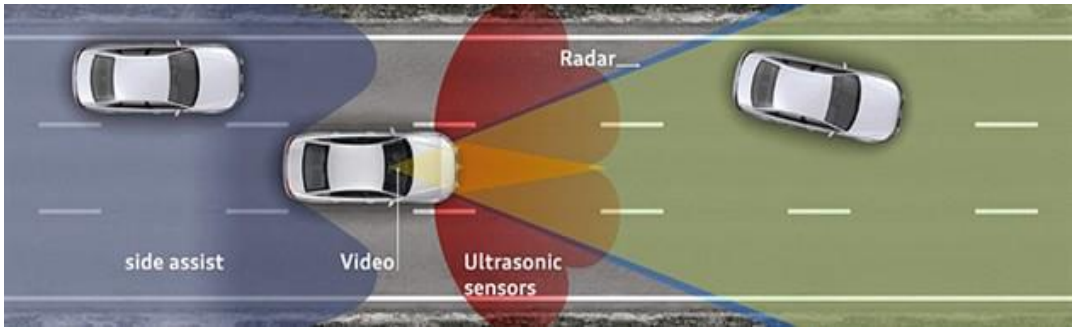


Figura 5: Adaptative Cruise Control (ACC)

Algunas de las funciones de los sistemas ACC más avanzados son las siguientes [7]:

-"Stop&Go".

Mediante un sensor los sistemas ACC más avanzados permiten que el vehículo, de forma automática, disminuya la velocidad al detectar un coche que circula más a una velocidad menor por delante y por contrario que recupere la marcha inicial programada cuando éste ya no se encuentre en la trayectoria.

- Reconocimiento de límites de velocidad.

El sistema de detección de señales de tráfico mediante cámaras es otra de las funciones del ACC avanzado. El conductor recibe la información sobre la velocidad máxima permitida por medio de voz y visual a través del cuadro de instrumentos. Como opción, el coche también adapta su velocidad a la máxima que indica la señal de tráfico.

- Control para mantener la trayectoria.

Gracias a un sistema de reconocimiento de imágenes, es posible detectar de forma fiable los bordes del carril. En esta función, una unidad electrónica calcula la distancia entre el coche y las líneas del carril. En el futuro, un aviso acústico o una vibración del volante alertará al conductor en caso de abandonar la trayectoria.

- Control de velocidad en curva.

Una de las principales limitaciones de los actuales sistemas ACC se presenta al trazar curvas cerradas. En estos casos, el coche que marcha por delante puede quedar fuera del campo de visión del sensor y éste también puede reconocer un vehículo que circula en sentido contrario y que no es relevante en nuestra trayectoria. Estos inconvenientes se solventan mediante los dispositivos de reconocimiento de las líneas del carril y el sistema de navegación predictiva. Gracias a los primeros, el coche reconoce continuamente los bordes de la carretera, mientras que el segundo permite ajustar la velocidad del vehículo a las

condiciones de la carretera mediante una serie de parámetros adicionales incluidos en el CD de navegación.

Una limitación de los sistemas ACC es que cuando la detección del radar se ve entorpecida por lluvia, nieve o polvo, la regulación de la distancia y la velocidad se desactivan automáticamente. En cuanto desaparece la circunstancia que impide la visibilidad, el conductor puede volver a activar la regulación del ACC.

2.3.2. Sistema antibloqueo de frenos (ABS)

El ABS (Antilock Braking System, sistema de antibloqueo de frenos) es un sistema que se utiliza para evitar que los neumáticos pierdan la adherencia con el suelo durante un proceso de frenado.

El sistema fue desarrollado inicialmente para los aviones, los cuales acostumbran a tener que frenar fuertemente una vez han tomado tierra. En 1978 Bosch hizo historia cuando introdujo el primer sistema electrónico de frenos antibloqueo. Esta tecnología se ha convertido en la base para todos los sistemas electrónicos que utilizan de alguna forma el ABS, como por ejemplo los controles de tracción y de estabilidad.

El ABS funciona en conjunto con el sistema de frenado tradicional. Consiste en una bomba que se incorpora a los circuitos del líquido de freno y en unos detectores que controlan las revoluciones de las ruedas. Si en una frenada brusca una o varias ruedas reducen repentinamente sus revoluciones, el ABS lo detecta e interpreta que las ruedas están a punto de quedar bloqueadas sin que el vehículo se haya detenido. Esto quiere decir que el vehículo comenzará a deslizarse sobre el suelo sin control, sin reaccionar a los movimientos del volante. Para que esto no ocurra, los sensores envían una señal al Módulo de Control del sistema ABS, el cual reduce la presión realizada sobre los frenos, sin que intervenga en ello el conductor. Cuando la situación se ha normalizado y las ruedas giran de nuevo correctamente, el sistema permite que la presión sobre los frenos vuelva a actuar con toda la intensidad. El ABS controla nuevamente el giro de las ruedas y actúa otra vez si éstas están a punto de bloquearse por la fuerza del freno. En el caso de que este sistema intervenga, el procedimiento se repite de forma muy rápida, unas 50 a 100 veces por segundo, lo que se traduce en que el conductor percibe una vibración en el pedal del freno.

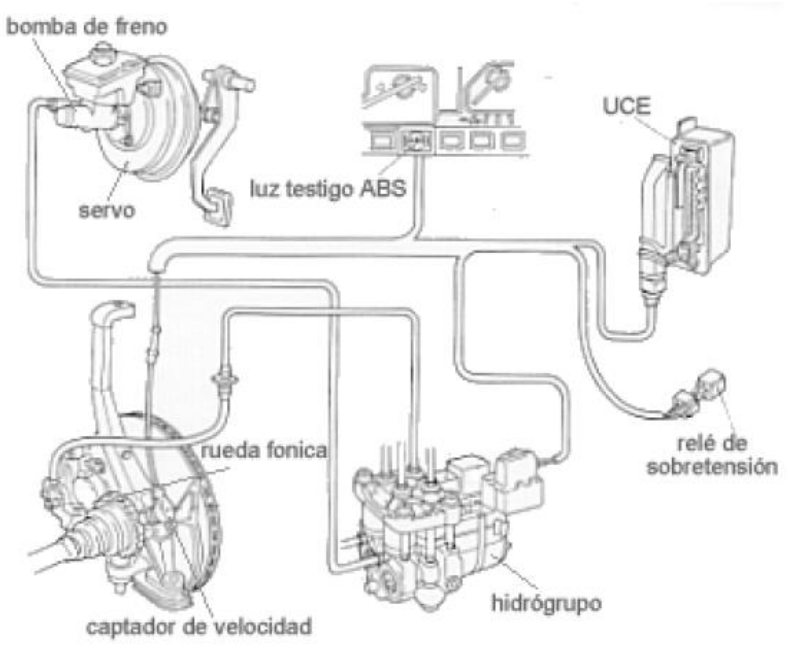


Figura 6: Esquema de elementos del ABS

2.3.3. Control electrónico de estabilidad (ESC)

El control de estabilidad es un elemento de seguridad activa del automóvil que actúa frenando individualmente las ruedas en situaciones de riesgo para evitar derrapes, tanto sobrevirajes, como subvirajes. El control de estabilidad centraliza las funciones de los sistemas ABS,EBD y de control de tracción.

El control de estabilidad (salvo que se desconecte manualmente) está activado permanentemente. En él, un microordenador evalúa las señales de los sensores y comprueba 25 veces por segundo si las maniobras del conductor al volante se corresponden con el movimiento real del vehículo. Si éste se mueve en una dirección diferente a la deseada, el ordenador detecta esta situación crítica y reacciona de inmediato, independientemente del conductor. El ESC utiliza el sistema de frenos, decelerando independientemente cada rueda para mantener estable la trayectoria del vehículo. Con este frenado selectivo el control de estabilidad genera la necesaria fuerza opuesta, de manera que el vehículo obedece al conductor. El sistema también puede intervenir en el motor para reducir la potencia del mismo. De esta manera, siempre dentro de los límites de la física, el vehículo mantiene con seguridad la trayectoria deseada.

El sistema está compuesto por:

- Un grupo hidráulico y unidad de control integrada (ECU), el grupo hidráulico ejecuta las órdenes de la unidad de control y regula mediante válvulas la presión de frenado de cada rueda. Además como

comentamos anteriormente, la ECU tiene comunicación constante con la gestión del motor para reducir la potencia en caso necesario.

- Cuatro sensores de velocidad en rueda, estos sensores son los mismos que los usados en el sistema antibloqueo de frenos o ABS, y son los encargados medir sin roce y mediante campos magnéticos, la velocidad de cada rueda. Los sensores de velocidad pueden ser pasivos o activos. Actualmente se usan casi siempre sensores activos, ya que permiten un mayor registro de velocidad (funcionan a partir de 0 km/h), son digitales, más precisos y pueden detectar también el sentido de giro. Estos sensores se basan en los principios magneto-resistivos o de efecto Hall.

- Un sensor de ángulo de dirección, situado en la columna de dirección mide, sin contacto, la posición del volante, determinando el ángulo de la dirección al conducir. En base a esta posición, a la velocidad del vehículo y a la presión de los frenos deseada o posición del pedal de acelerador se calcula la intención de la maniobra deseada por el conductor. Los primeros sensores de ángulo eran de tipo incremental y median pulsos relativos a la posición del eje. Aunque en la actualidad son de tipo absoluto, que produce un código digital único para cada ángulo distinto del eje. Al igual que los sensores de velocidad en rueda son magneto-resistivos o de efecto Hall.

- Un sensor de ángulo de giro y aceleración transversal: es en realidad dos sensores en uno, proporciona información sobre desplazamientos del vehículo alrededor de su eje vertical, desplazamientos y fuerzas laterales, es decir, cual es el comportamiento real del vehículo, si está comenzando a derrapar y desviándose de la trayectoria deseada por el conductor. Este sensor suele estar situado en el centro del vehículo y funciona como una giróscopo y acelerómetro de tres ejes combinados.

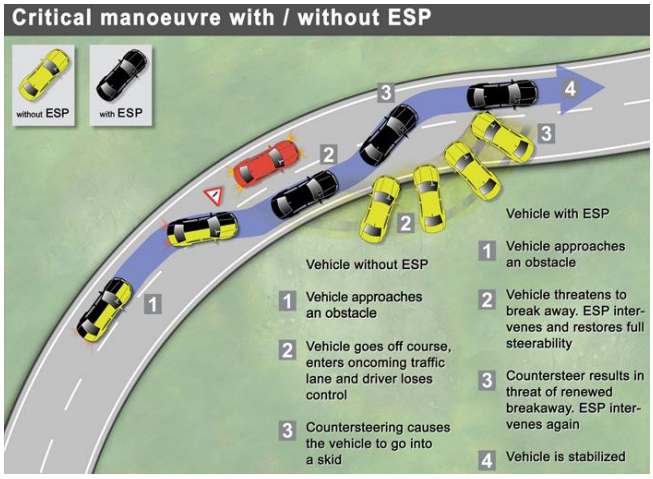


Figura 7: Trayectoria con/sin ESC

2.3.4. Sistemas de mejora de la visibilidad.

Ante una curva, los faros adaptativos orientan el haz de luz, proporcionando al conductor una mejor imagen de lo que tiene delante. Los sistemas de visión nocturna mejoran considerablemente la visibilidad en condiciones de poca luminosidad en comparación con los faros tradicionales, ya que emplean una tecnología de captación de imágenes nocturnas.

Figura 8: Sistemas de mejora de la visibilidad.

Como fabricante a destacar en este tipo de sistema es Opel. El sistema Adaptive Forward Lighting (AFL) incluye dos funciones de seguridad principales: las luces de curva dinámicas y las luces de curva estáticas. La función de luces de curva dinámicas ilumina la parte de la carretera hacia la que el vehículo se dirige orientando los conos de los faros hacia el interior de la curva. La función de luces de curva estáticas son unas luces de giro adicionales que se encienden para ayudar al conductor en curvas muy cerradas.

2.3.5. Sistemas de detección de ángulo muerto.

Es un error muy común cambiar de carril cuando hay un vehículo en el "punto ciego" (detrás y a un lado del vehículo). Esta maniobra provoca muchos accidentes en las autopistas europeas.

Diversos fabricantes han desarrollado sistemas que controlan el punto ciego y ayudan a un conductor a cambiar de carril con seguridad. Hay sistemas que cuentan con cámaras de vídeo y otros que basan su funcionamiento en un radar. En cualquier caso, estos sistemas controlan la zona de la esquina trasera del vehículo y avisan al conductor de la aproximación de un vehículo que, con los retrovisores, habría sido imposible detectar.

Una luz indicadora en el retrovisor de la puerta indicará al conductor de la presencia de un vehículo en el punto ciego del lado correspondiente. Si el conductor acciona el intermitente cuando un vehículo se encuentra en la zona de peligro, el indicador de advertencia empezará a parpadear y emitirá una señal sonora.

Con el objetivo de minimizar las falsas alarmas, generalmente actúan por encima de un umbral de velocidad determinado y son capaces de realizar un filtrado de vehículos estacionados o de aquellos que circulan en sentido contrario. La zona de detección es unos 10 metros por detrás del espejo retrovisor por unos 4 de anchura, suficiente para cubrir el ángulo muerto.

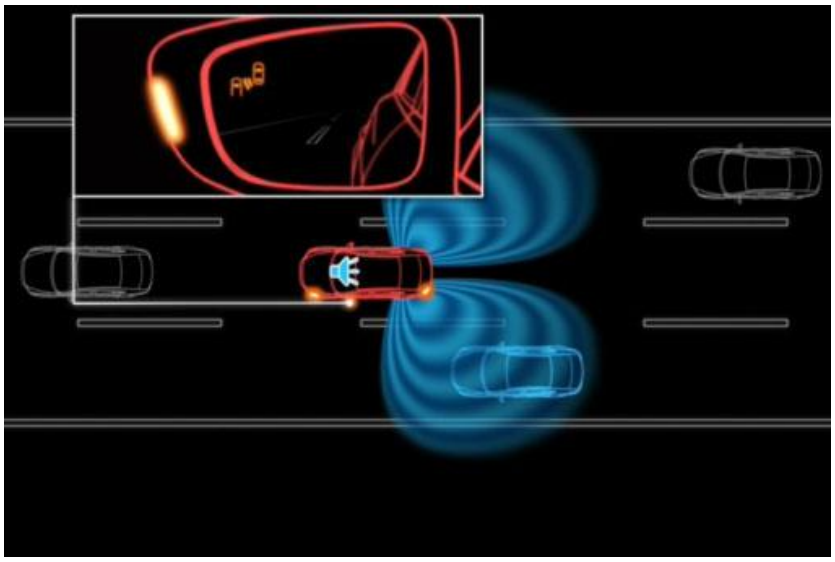


Figura 9: Sistema de control del punto ciego

Algunos fabricantes que han desarrollado este tipo de sistema son:

- Opel Side Blind Spot Assistance (SBSA). Está basado en sensores ultrasónicos.
- Mazda Rear Vehicle Monitoring (RVM). Basado en dos sensores de radar para medir la distancia y la velocidad relativa de los vehículos cercanos.
- Audi Side Assist. Basado en dos radares colocados en la parte trasera del vehículo.

2.3.6. Asistente para cambios de carril

Es un sistema que amplía las posibilidades de funcionamiento del sistema de detección de objetos en ángulo muerto.

Los sistemas de asistencia de cambio de carril le avisan al conductor cuando abandona de forma involuntaria el carril o cuando cambia de carril sin indicarlo. El sistema controla la posición del vehículo y, mientras el aviso de cambio de carril le advierte de que ha abandonado de forma involuntaria la trayectoria, el sistema de permanencia en el carril le ayuda a corregir la trayectoria del vehículo.

Cada sistema utiliza un aviso distinto como una señal acústica o una vibración en el volante para simular la sensación de que el vehículo está circulando sobre "bandas sonoras".

El objetivo es hacer que el conductor se percate de que el vehículo corre el peligro de cruzar la línea. Algunos sistemas solo necesitan una línea en un lado del vehículo, mientras que otros requieren la existencia de marcas delimitadas a ambos lados. La mayoría de los sistemas solo funcionan cuando se alcanzan velocidades propias de una autopista y desactivan la señal de aviso si se utiliza el intermitente.

Se coloca una cámara detrás del espejo retrovisor. Un ordenador analiza las imágenes de esta cámara de forma continua para identificar las marcas de delimitación del carril y, en algunos casos, los límites no marcados de la carretera. Además, también se registran los movimientos del volante, así como la velocidad y la trayectoria del vehículo. Estos parámetros se combinan para determinar si el vehículo está o no abandonando el carril por el que transita.

El sistema solo funcionará en aquellos casos en los que las líneas se vean con claridad. En caso contrario se avisará al conductor para que sepa que el sistema no puede ayudarle.

Otro sistema con el mismo objetivo es el sistema de permanencia de carril. El sistema de permanencia en el carril desplaza el vehículo para que

regrese al carril. Cuando el vehículo se aproxima a la línea de delimitación, el sistema lo aparta de dicha línea hasta que vuelve a estar dentro del carril.

Algunos fabricantes que han desarrollado este tipo de sistema son:

- Skoda Lane Assistant.
- Audi Active Lane Assist.
- Seat Lane Assist.

El Audi Q7 fue el pionero en la utilización de un sistema de asistencia al cambio de carril, mediante el uso de dos radares de medio alcance ubicados en las esquinas del paragolpes posterior.

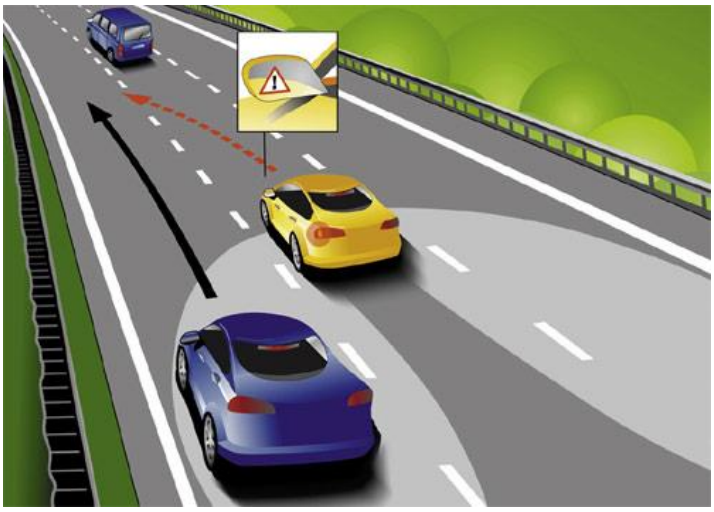


Figura 10: Asistente de cambio de carril

2.3.7. Llamada automática de emergencia (eCall).

La llamada automática de emergencia (eCall) es un sistema que envía un mensaje automático, que contiene la ubicación del vehículo indicada por un dispositivo GPS integrado, a un centro de llamadas de emergencia cuando el vehículo se ve involucrado en una colisión.

Entre los fabricantes que han desarrollado este tipo de sistema destacan Ford y BMW.

- El sistema de Ford (SYNC Emergency Assistance) utiliza el teléfono móvil del usuario cuando está conectado por Bluetooth al sistema SYNC. Cuando el vehículo sufra un accidente lo bastante grave como para desplegar los airbags, el Asistente de emergencia llamará automáticamente al 112 y enviará un mensaje para informar de que un vehículo Ford ha sufrido un accidente y proporcionar las coordenadas GPS de la ubicación.

- El sistema BMW Assist Advanced eCall es un sistema automático de llamadas de emergencia que predice la gravedad de las lesiones de los ocupantes. Si los sensores de colisión detectan que un vehículo se ha visto implicado en un accidente, Advanced eCall se pone en contacto automáticamente con el centro de atención de llamadas de BMW y proporciona información detallada sobre el accidente. También predice el riesgo de lesiones graves con un algoritmo basado en conocimientos conocido como URGENCY. Se envía un mensaje SMS a un centro de atención de llamadas de BMW con la información más importante. Además, se establece una conexión de voz directa entre el vehículo y el centro de atención de llamadas. La información que proporcionen los ocupantes del vehículo, junto con los datos del SMS, se utilizará para determinar la respuesta más apropiada a la emergencia. Si los ocupantes del vehículo están inconscientes o no pueden comunicarse, se avisará automáticamente a los servicios de emergencia.

Figura 11: Sistema eCall

2.3.8. Sistemas de alerta por pérdida de atención.

El sistema de alerta por pérdida de atención es una función que avisa a los conductores para que no se queden dormidos mientras conducen. Les indica que se tomen un descanso antes de que sea demasiado tarde.

Los principales fabricantes que han desarrollado este tipo de sistema son Ford y Mercedes-Benz.

- Ford Driver Alert. Es un asistente para el conductor que emplea una cámara orientada hacia delante para supervisar la posición del vehículo en el carril y calcula el nivel de atención del conductor. Si el

nivel de atención desciende por debajo de un valor, el vehículo emite una advertencia sonora y visual.

- Mercedes-Benz Attention Assist. Este sistema emplea un sensor de ángulo de la dirección para determinar la forma en la que el conductor controla el vehículo. A velocidades comprendidas entre los 80 y los 180 km/h, el sistema identifica un patrón de conducción propio de un conductor cansado y combina esta información con otros datos como la hora del día y la duración del trayecto. Si el sistema identifica estas circunstancias, avisa al conductor de que debe tomar un descanso mediante un testigo con forma de taza de café en el panel de instrumentos y un aviso sonoro.

Figura 12: Sistema de alerta por pérdida de atención

2.3.9. Sistemas de alerta de velocidad (ISA).

Los sistemas inteligentes de control o alerta de velocidad (ISA) ayudan a los conductores a mantener la velocidad dentro de los límites recomendados.

Algunos sistemas muestran el límite actual para que el conductor sea siempre consciente de la velocidad máxima permitida en esa carretera. El límite de velocidad puede determinarse con un software que analiza las imágenes de una cámara y reconoce las señales de tráfico. Asimismo, la navegación por satélite se ha vuelto extremadamente precisa y podría utilizarse para proporcionar información al conductor. Sin embargo, es necesario contar con mapas digitales actualizados en todo momento. Los sistemas podrían o no emitir un aviso cuando el límite de velocidad se supere.

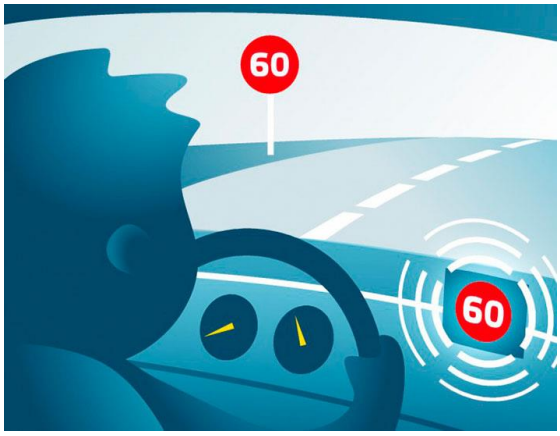


Figura 13: Sistema de alerta de velocidad (ISA)

2.3.10. Sistemas de visión nocturna.

De noche o cuando las condiciones meteorológicas son adversas se reduce la visibilidad y, a veces, los faros no proporcionan la iluminación suficiente, por lo que la circulación se vuelve más compleja y peligrosa, sobre todo cuando se conduce por vías no urbanas.

Estos sistemas de visión nocturna muestran a través de una pantalla situada en la consola lo que sucede en el campo de movimiento del vehículo cuando las condiciones de iluminación son reducidas, disminuyendo el número de atropellos y accidentes.

En la actualidad, estos sistemas se ofrecen como equipamiento opcional en las gamas altas de determinadas marcas de vehículos.

Dependiendo de la tecnología utilizada existen dos tipos de sistemas: sistemas activos y sistemas pasivos. Los sistemas activos emplean dispositivos de luz infrarroja, instalados en los automóviles. Los sistemas pasivos, por otro lado, captan la radiación térmica emitida por los objetos mediante el empleo de una cámara termográfica. La ventaja de este último sistema frente al primero es que transmite únicamente la información más importante, evitando distraer al conductor.

Figura 14: Sistema de visión nocturna

2.3.11. Sistemas de identificación de señales.

Esta tecnología consiste en detectar las señales de tráfico de la carretera e identificarlas para informar al conductor de la prohibición o advertencia que muestran las mismas.

Consiste en una cámara que capta las imágenes de la carretera y mediante un procesado de imagen digital realizado por un ordenador es capaz de identificar la señal en la imagen e interpretarla para mostrarla al conductor a través de un interfaz situado en el salpicadero, como se muestra en la *Figura 15*.

Existe otro tipo de sistema más avanzado que permite al coche intervenir dependiendo de la orden que muestre la señal detectada.

Figura 15: Sistema de identificación de señales.

2.3.12. Sistema de prevención de colisiones.

Estos sistemas están destinados a optimizar el funcionamiento y la eficacia de los sistemas de protección del vehículo durante un impacto. Algunos de estos sistemas reaccionan de forma inmediata durante el impacto o tras haber ocurrido con el fin de optimizar la seguridad de los ocupantes. Otros sistemas pueden predecir cuándo va a ocurrir un accidente y preparar el vehículo y sus ocupantes para la colisión.

Las acciones que normalmente realizan son ajustar los cinturones de seguridad, fijar las posiciones de los asientos para optimizar el rendimiento de los airbag y cerrar las ventanillas para evitar que los ocupantes salgan despedidos. En esos casos, las acciones realizadas se invertirán si el accidente consigue evitarse.

Destacan en este tipo de sistema los fabricantes Skoda y Audi:

- Skoda Crew Protect Assist. Sistema que detecta si se realiza una maniobra de emergencia, y prepara el vehículo y los sistemas de sujeción de los ocupantes ante una posible colisión. A velocidades superiores a los 30 km/h, el sistema activará los pretensores eléctricos para tensar los cinturones de seguridad delanteros. Si no se produce una colisión, los pretensores eléctricos se desactivarán y la conducción volverá a un estado normal. Los pretensores tienen dos niveles de tensión: un bajo que se puede desactivar y uno alto que siempre se activará en situaciones críticas.

- Audi Pre-Sense Basic. Mismo sistema que el de Skoda pero incluye además el cierre de las ventanillas y el techo solar. Esto evitará que, en caso de colisión lateral, entren objetos extraños en el habitáculo.

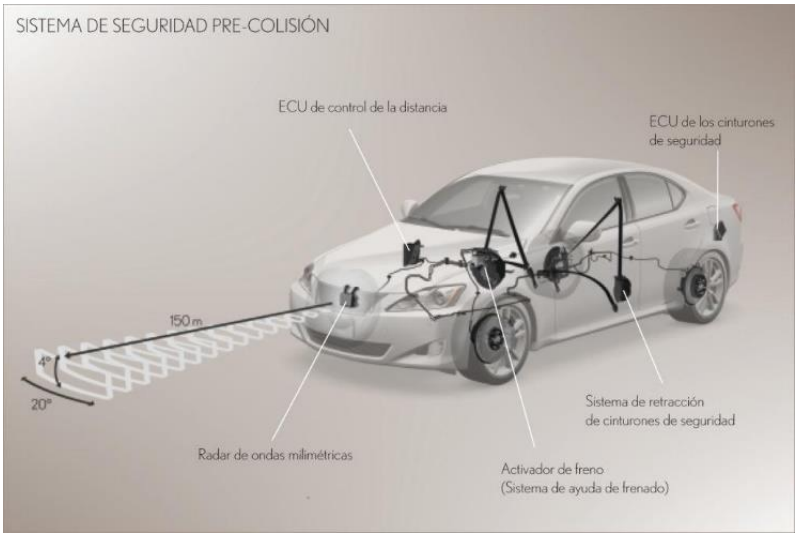


Figura 16: Sistemas precolisión.

2.4. Programa Europeo de Evaluación de Automóviles Nuevos (EuroNCAP).

Euro NCAP ("Programa Europeo de Evaluación de Automóviles Nuevos") es un programa que ofrece información sobre los niveles de seguridad de diferentes modelos de vehículos que se evalúan a partir de los resultados obtenidos en una serie de ensayos y pruebas de seguridad activa y pasiva. Ofrece a los consumidores una evaluación realista e independiente del comportamiento de la seguridad del automóvil comercializado en Europa.

Fundada en 1997 a partir de la unión de la Federación Internacional del Automóvil y la Administración Nacional de Vialidad de Suecia.

En 1998 Euro NCAP siguió el estatus legal cuando se convirtió en asociación internacional bajo la ley belga.

En 2009, se añadió a las pruebas existentes la prueba de choque por detrás, donde se evaluará los daños a los pasajeros en el cuello y la espalda, y se comprobará el funcionamiento de algunos elementos de seguridad activa como el control de estabilidad o el limitador de velocidad. También se modificó el modo de calificación, que abarcará las cuatro áreas de pruebas realizadas a un modelo.

Con el apoyo actual de siete gobiernos europeos y las organizaciones de automovilismo y consumidores de cada país de la UE, Euro NCAP se ha convertido rápidamente en un catalizador para fomentar importantes mejoras de seguridad en el diseño de vehículos nuevos.

Euro NCAP es por sí sola una Asociación Internacional de jurisdicción belga. Es independiente de la industria y políticas de control, y ningún miembro individual puede influir en Euro NCAP a favor de sus intereses individuales. Es totalmente independiente de la industria del automóvil.

Inicialmente, los fabricantes de automóviles se posicionaron en contra del programa Euro NCAP. Sin embargo, percibieron que recibía mucha publicidad y que se consideraba técnicamente correcto. Los vehículos que recibieron valoraciones positivas han mejorado sus ventas y los que obtuvieron malos resultados perdieron volumen de ventas.

Al ver las ventajas de tener una evaluación independiente que guíe el diseño de seguridad del coche en la dirección correcta, los fabricantes de automóviles comenzaron a apoyar el programa ya que se realiza de una forma objetiva y es el responsable de la mejora de las normas de seguridad globales.

Los fabricantes participan en los ensayos. Se informa a cada fabricante de la elección del coche, variantes y opciones. Se pide a los fabricantes que proporcionen información sobre la configuración de los ensayos y hagan observaciones generales. Los fabricantes presencian los ensayos y comentan si están satisfechos con la forma en la que se desarrolla el ensayo. Después del mismo, reciben los resultados comentan la posible existencia de cualquier anomalía en comparación con sus propios datos.

Los fabricantes de vehículos tienen que diseñar y someter a ensayo sus vehículos con el fin de cumplir con muchas más cuestiones de seguridad que las que se ponen a prueba en los ensayos de Euro NCAP

Los informes de los ensayos indican las zonas donde una mejora proporcionaría beneficios de seguridad. Si los fabricantes quieren tomar una decisión sobre la base de los resultados, son ellos los que deciden cambiar el diseño de sus productos o aplicar las mejoras. Euro NCAP no es quien tiene que decir a los fabricantes cómo diseñar sus coches.

Por ley, todos los nuevos modelos de coches deben superar ciertos ensayos de seguridad antes de ponerse a la venta. De este modo, la legislación proporciona un estándar mínimo de seguridad obligatorio para los coches nuevos.

El objetivo de Euro NCAP consiste en exigir a los fabricantes a superar unos estándares mínimos de seguridad.

Figura 17: Logotipo Euro NCAP

3. Fundamentos teóricos.

En el presente capítulo se van a introducir los conceptos teóricos mínimos que son necesarios para el desarrollo del algoritmo de detección de obstáculos.

En primera instancia se hablará sobre la óptica donde se hará referencia a los modelos principales a la hora de obtener una imagen mediante una cámara estéreo. Puesto visión estereoscópica. Finalmente se desarrollará conceptualmente y matemáticamente los descriptores HOG y las Máquinas de Soporte Vectorial.

3.1. Óptica.

Para tener una mejor concepción de las lentes se van a explicar los dos modelos principales que nos permiten captar una imagen. Estos dos modelos son el modelo de lente fina y el modelo pin-hole.

3.1.1. Modelo de lente fina.

Este modelo consiste en una lente biconvexa de grosor despreciable que permite concentrar en un punto los infinitos rayos luminosos procedentes de un punto del espacio. Todos los rayos paralelos al eje óptico convergen en un punto del eje óptico a una distancia igual a la distancia focal del centro de la lente. Los rayos que atraviesan el centro óptico no se desvían.

Este modelo se basa en la fórmula de Gauss de las lentes delgadas con rayos paraxiales, y está diseñado para mostrar la formación de imágenes mediante rayos.

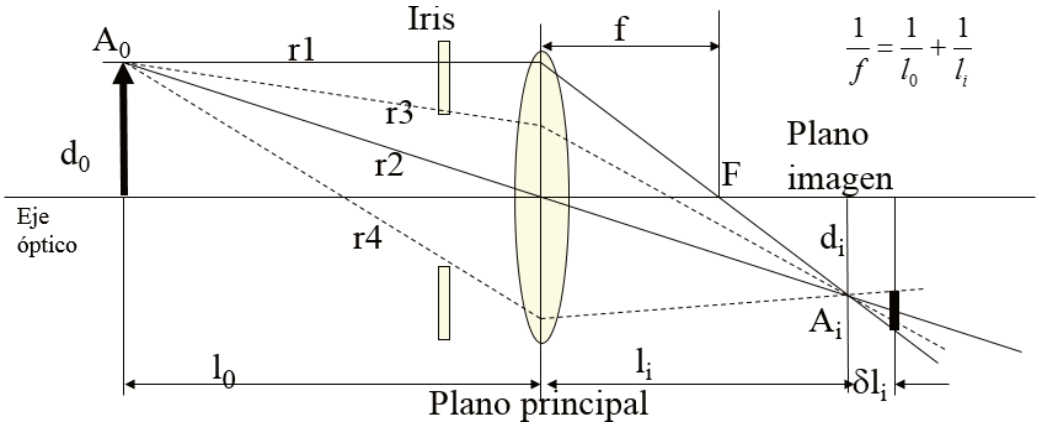


Figura 18: Modelo de lente fina.

3.1.2. Modelo Pin-Hole.

Es el modelo más utilizado. Una cámara pin-hole es una cámara muy simple que no tiene lente y que tiene una sola y muy pequeña apertura, de ahí su nombre, pin-hole, que es la traducción al inglés de la palabra “hoyo de

aguja". Esta cámara puede ser vista de otra manera como una caja a prueba de luz con un pequeño orificio en uno de sus lados, siendo ese orificio la única entrada de luz que permite la caja. Cuando la luz de una imagen pasa a través de este agujero entonces se forma una imagen invertida en el lado opuesto de la caja.

El ojo humano trabaja de la misma forma y como estamos familiarizados con la explicación del funcionamiento del ojo es posible generarse una mejor idea del funcionamiento de la cámara pin-hole de esta manera.

En las cámaras pin-hole, mientras más pequeño es el agujero por el que atraviesa la luz, más nítida y definida es la imagen que se forma en el interior de la misma. Sin embargo, la imagen se hace también más oscura conforme el orificio disminuye de tamaño. Como conclusión, cuanto más definida es la imagen, más opaca será.

Para tener un buen funcionamiento se recomienda que el orificio sea del 1% o menos de la distancia entre el orificio y la pantalla.

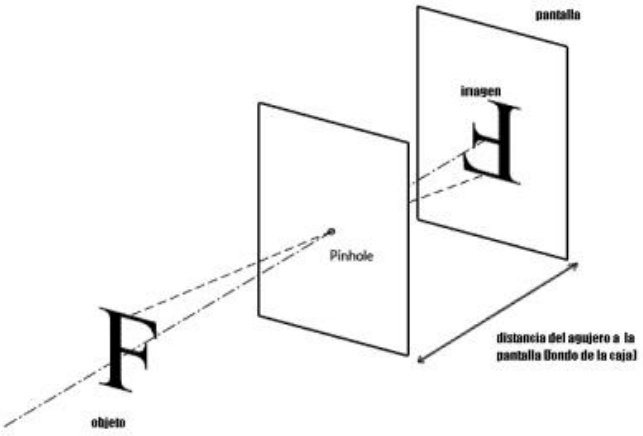


Figura 19: Cámara pin-hole

El modelo pin-hole describe la relación existente entre las coordenadas de un punto en 3D y su proyección en el plano de la imagen. Las ecuaciones (1) y (2) se obtienen por triangulación a partir de los parámetros que se indican en la *Figura 20*.

$$x = \frac{f}{z} \cdot X \tag{1}$$

$$y = \frac{f}{z} \cdot Y \tag{2}$$

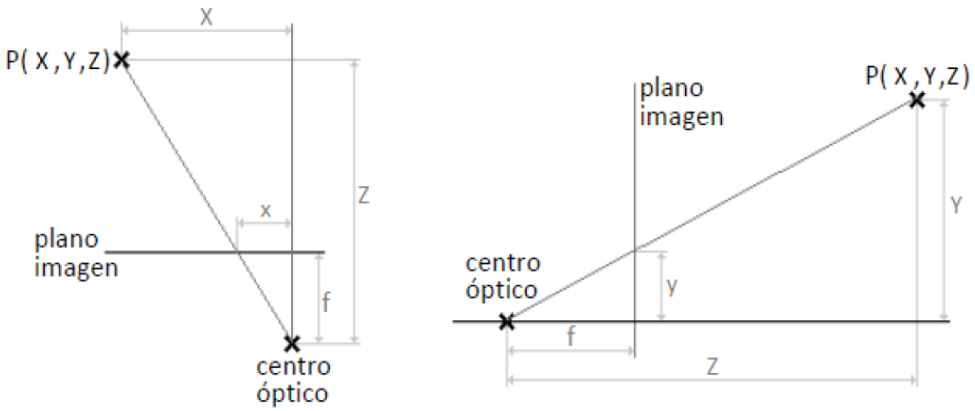


Figura 20: Esquema del modelo pin-hole.

3.1.3. Visión estéreo.

Un sistema de visión estéreo se compone de dos cámaras. Para tener un modelo ideal, los ejes ópticos de cada cámara deben ser paralelos y además tener la misma distancia focal.

El procedimiento consiste en captar dos imágenes de una misma escena, cada imagen es capturada desde una posición de las cámaras ligeramente diferente, por lo que las imágenes se presentan también ligeramente desplazadas entre sí, siendo éste el fundamento básico de la visión estereoscópica, ya que este hecho es el que va a permitir la obtención de la distancia a la que se encuentra un determinado objeto.

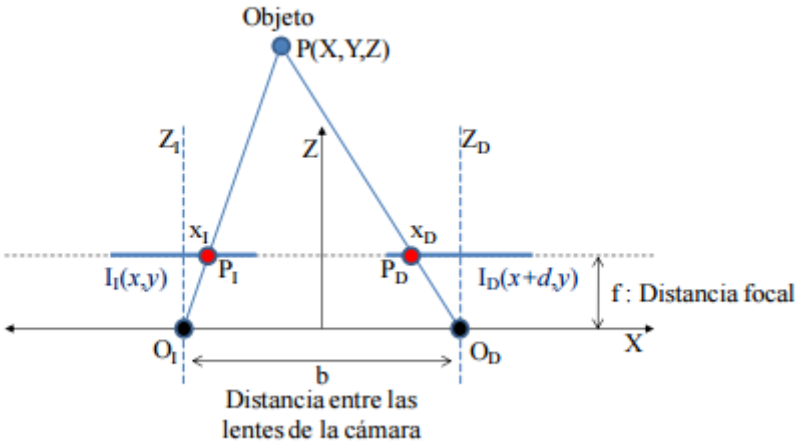


Figura 21: Representación de la proyección estereoscópica.

Considerando una relación geométrica de semejanza de triángulos, las coordenadas del punto de la escena *P (X, Y, Z)* pueden deducirse fácilmente sin más que observar la *Figura 20*, obteniendo los resultados dados por la ecuación

de la *Figura 21*. Se deduce a partir de la ecuación que cuando se utiliza esta geometría, la profundidad *Z*, es inversamente proporcional a la disparidad de la imagen y para una profundidad dada, a mayor *b* mayor *d*.

$$\left.\begin{array}{l} O_I : \dfrac{\frac{b}{2}+X}{Z}=\dfrac{x_I}{f} \\ O_D : -\dfrac{\frac{b}{2}-X}{Z}=\dfrac{x_D}{f} \end{array}\right\} \Rightarrow \left.\begin{array}{l} x_I = \dfrac{f}{Z}\left(X+\dfrac{b}{2}\right) \\ x_D = \dfrac{f}{Z}\left(X-\dfrac{b}{2}\right) \end{array}\right\} \Rightarrow \mathrm{d} = x_I - x_D = \frac{fb}{Z} \Rightarrow Z = \frac{fb}{d}$$

Figura 22: Ecuación de la visión estereoscópica.

3.2. Descriptores basados en Histogramas de Gradientes Orientados.

Los descriptores HOG (Histogram of Oriented Gradients-HOG) describen la orientación del gradiente en toda una región dada. Estos descriptores calculan un espacio de celdas superpuestas con el fin de normalizar las muestras locales y aumentar la precisión. La imagen se divide en celdas las cuales acumulan los histogramas de orientación de gradientes de las celdas. Estos histogramas capturan propiedades (bordes) de forma local dentro de la celda. A su vez, estos histogramas son invariantes a pequeñas deformaciones. El gradiente en cada píxel está discretizado en uno de los nueve contenedores (bins) de orientación, y cada píxel vota por la orientación de su gradiente, con un valor que depende de la magnitud del gradiente.

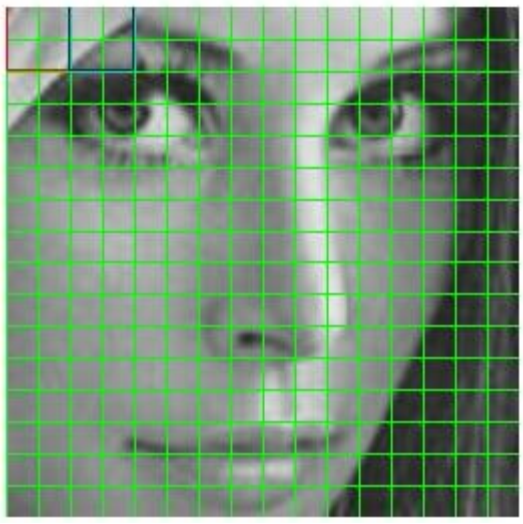


Figura 23: Bloque 1 (azul), bloque 2 (rojo), celdas (verde).

Los descriptores HOG nos permiten conocer los cambios de intensidad debido a los contornos o bordes de una imagen. Al tener en cuenta la relación con sus zonas vecinas y colindantes, es posible reconocer cuándo existe una frontera entre un objeto y otro. De esta manera, podremos identificar objetos de siluetas más suaves o más pronunciadas.

El descriptor HOG está adecuado para la detección de cualquier obstáculo, independientemente de su tamaño y color.

3.2.1. Idea principal.

Las bases teóricas de los métodos HOG residen en trabajos previos tales como Histogramas de bordes orientados (Freeman and Roth 1995), descriptores SIFT (Lowe 1999) y reconocimiento de formas (Belongie et al. 2001), entre otros. Sin embargo, los métodos HOG presentan una diferencia añadida que consiste en que los gradientes no se calculan uniformemente sobre un mallado denso, sino que divide la imagen en bloques y, a su vez, cada bloque en sub-bloques, y se calcula en cada uno de ellos los gradientes y el histograma.

El cálculo de los descriptores HOG presenta un coste de tiempo de computación bajo por el hecho de calcular el HOG en cada una de las celdas.

La elección de este método para la detección de motocicletas en la escena se basa en que destaca por su robustez frente a diferentes condiciones de iluminación, pequeños cambios en el contorno de la imagen y diferencias de fondos y de escalas.

En caso de tener una imagen en color, lo primero que se debe hacer es transformarla a escala de grises. Posteriormente se calculan los gradientes espaciales sobre toda la imagen. Más tarde, se divide la imagen en bloques, con cierta área solapada. El avance de bloques en el eje horizontal se realiza eliminando la columna de las celas de la izquierda y añadiendo la columna de la derecha, mientras que para el eje vertical se elimina la fila de las celdas de arriba añadiendo la fila de celdas de abajo.

Finalmente se aplica una ventana gaussiana sobre cada bloque, almacenándose dicha información en el vector de características de la imagen.

En la *Figura 24* se muestra el procedimiento a seguir para el cálculo de los descriptores de una imagen de dimensiones de 64 píxeles de ancho y 128 píxeles de alto, es decir, 64x128.

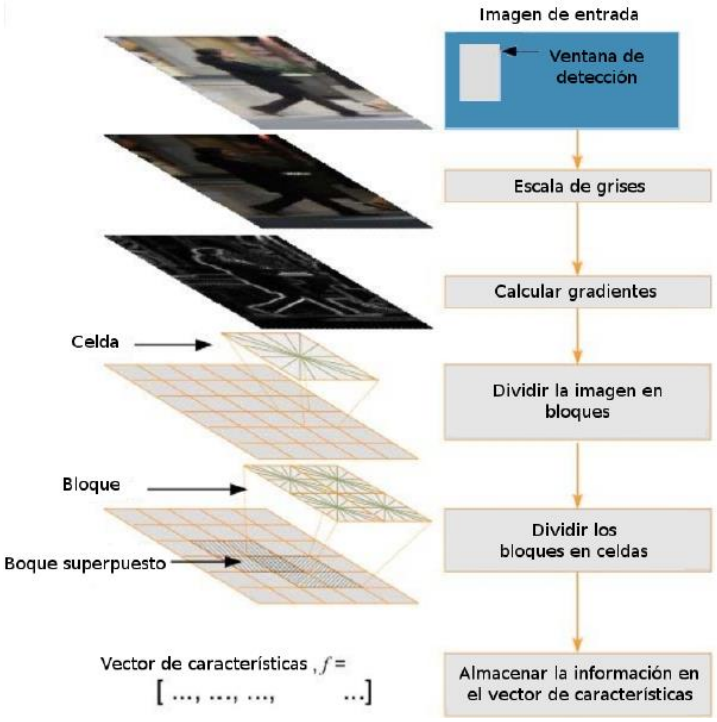


Figura 24: Proceso de extracción de características.

3.2.2. Fundamento matemático.

El histograma de una imagen divide el rango de valores de los píxeles de la imagen en una serie de sub-rangos o clases (p. ej. dado el rango de valores de píxel [0, 255], se divide en ocho clases de un mismo tamaño: [0,32), [32,64), [64,96), (96,128],… [224, 255]), y almacena la frecuencia de píxeles de cada clase con un valor comprendido en ese sub-rango, es decir, el número de píxeles en la imagen cuyos valores están entre los valores de inicio y fin de cada sub-rango.

El HOG de una imagen tiene como rango de valores las distintas orientaciones que pueden tomar los gradientes de los píxeles, es decir, los distintos grados que pueden tomar sus ángulos de gradiente (p. ej. [-90°, 90°], [0°, 180°], [0°, 360°],…). Este rango se divide en sub-clases (del mismo o distinto tamaño), (ej. Para el rango [0°, 180°], se puede dividir en nueve sub-rangos: [0°, 20°), [20°, 40°), [40°, 60°), [60°, 80°), [80°, 100°), [100°, 120°), [120°, 140°), [140°, 160°), [160°, 180°]), y se almacena en cada uno de ellos la suma de las magnitudes de gradiente de los píxeles cuyo ángulo de gradiente se encuentra comprendido entre esos valores.

Partiendo del concepto de los Histogramas de Gradiente Orientados, se puede obtener más información sobre una imagen mediante un Descriptor HOG. En éste, la imagen se divide en un número de sub-imágenes del mismo tamaño, denominadas celdas, y éstas se agrupan en bloques con un mismo número de celdas de ancho y alto todos ellos. Además, se debe tener en

cuenta el solapamiento de los bloques y el avance de los mismos como ya se explicó en el apartado anterior.

De este modo, dados una imagen A de tamaño $W\ x\ H$, un tamaño de celda $C_W\ x\ C_H$, con $WmodC_W = 0$ y $HmodC_H = 0$; y con un tamaño de bloque en celdas $B_W\ x\ B_H$; el ancho y alto de la imagen en celdas, W_C y H_C, y el número de bloques distribuidos horizontalmente y verticalmente, N_{BW} y N_{BH}, se calculan con las siguientes ecuaciones:

$$W_C = \frac{W}{C_W}, \qquad H_C = \frac{H}{C_H}$$

$$N_{BW} = 1 + W_C - B_W, \qquad N_{BH} = 1 + H_C - B_H$$

Figura 25: Ecuaciones ancho y alto de las celdas y número de bloques.

Por tanto, el número total de celdas N_C y el número total de bloques N_B resultantes de la imagen será igual a:

$$N_C = W_C * H_C$$

$$N_B = N_{BW} * N_{BH}$$

Figura 26: Número total de celdas y bloques.

Y la distribución de celdas (C) y bloques (B) es la siguiente:

$$A = \begin{pmatrix} a_{00} & a_{10} & \cdots & a_{(W-1)0} \\ a_{01} & a_{11} & \cdots & a_{(W-1)1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{0(H-1)} & a_{1(H-1)} & \cdots & a_{(W-1)(H-1)} \end{pmatrix}$$

$$C = \begin{pmatrix} c_{00} & c_{10} & \cdots & c_{(W_C-1)0} \\ c_{01} & c_{11} & \cdots & c_{(W_C-1)1} \\ \vdots & \vdots & \ddots & \vdots \\ c_{0(H_C-1)} & c_{1(H_C-1)} & \cdots & c_{(W_C-1)(H_C-1)} \end{pmatrix}$$

$$B = \begin{pmatrix} b_{00} & b_{10} & \cdots & b_{(N_BW-1)0} \\ b_{01} & b_{11} & \cdots & b_{(N_BW-1)1} \\ \vdots & \vdots & \ddots & \vdots \\ b_{0(B_H-1)} & b_{1(B_H-1)} & \cdots & b_{(N_BW-1)(N_BH-1)} \end{pmatrix}$$

Figura 27: Distribución de celdas y bloques.

A partir de la estructura de la imagen, el descriptor HOG calcula de forma independiente el HOG de cada celda y cada bloque agrupa los HOGs de sus celdas correspondientes. Entonces, el número de HOGs que tiene un descriptor será:

$$N_{HOG} = B_W * B_H * N_B$$

Figura 28: Número de HOGs de cada descriptor.

Y si dividimos el HOG en n clases, dado que cada bloque contiene $B_W \; x \; B_H$ descriptores HOG, entonces el número total de valores Nv que se tendrá de la imagen A será:

$$N_V = n * N_{HOG}$$

Figura 29: Número total de valores.

Para el cálculo del modelo de detección, sobre el conjunto de imágenes del entrenamiento se debe calcular el descriptor de HOG de cada imagen, etiquetando cada descriptor como positivo si es una imagen de motocicleta (+1) o negativo si no lo es (-1).

El coste en tiempo de computación del cálculo de los descriptores de HOG puede ser bastante grande ya que se hace el cálculo de un HOG por cada celda. Para agilizar este proceso se puede usar una técnica denominada Integral de HOG, el cual mejora el coste de computación a cambio de aumentar el coste en memoria. Para el cálculo de la Integral de HOG se necesitan n matrices auxiliares de tamaño igual a la imagen de entrada A_{WxH}, donde n es el número de clases en el HOG. Cada una de estas matrices estará asociada a una única clase, es decir, a un único rango de ángulos de gradiente, y donde cada uno de sus píxeles cumple:

$$M(b,i,j) = \begin{cases} |G(i,j)|, & \theta(i,j) \in R(b) \\ 0, & eoc \end{cases}$$

Figura 30: Integral de HOG.

Donde b Є {0, 1, …, n – 1} indica la clase, i Є {0, 1, …, W – 1} indica la columna de la matriz, j Є {0, 1, …, H – 1} indica la fila de la matriz y R(b) es el rango de valores de ángulo asociado a la clase b.

3.3. Máquinas de soporte vectorial (SVM).

Las Máquinas de Vectores Soporte son estructuras de aprendizaje basadas en la teoría estadística del aprendizaje. Se fundamentan en la transformación del espacio de entrada en otro de dimensión superior (infinita) en el que el problema puede ser resuelto mediante un hiperplano óptimo de máximo margen.

Estos métodos están relacionados con problemas de clasificación y regresión.

Dado un conjunto de muestras de entrenamiento podemos etiquetar las clases y entrenar una SVM para generar un modelo que prediga la clase de la nueva muestra. Intuitivamente, una SVM es un modelo que representa a los puntos de la muestra en el espacio, separando las clases por un espacio lo más amplio posible. Cuando las nuevas muestras se ponen en correspondencia con el modelo, en función de la proximidad entre las demás muestras pueden ser clasificadas como pertenecientes a una clase u otra clase.

En definitiva, una SVM construye un hiperplano o un conjunto de hiperplanos en un espacio de dimensionalidad muy alta o casi infinita que se utiliza en problemas de clasificación o regresión. Teniendo una buena separación entre las clases permitirá una clasificación correcta.

3.3.1. Idea principal.

Las Máquinas de Soporte Vectorial (SVM) son un conjunto de algoritmos de aprendizaje supervisado empleados para la clasificación y la regresión. Teniendo un conjunto de muestras de entrenamiento se puede etiquetar las clases y entrenar una SVM para construir un modelo que pueda predecir una clase de una nueva muestra.

Tomando los datos de entrada como conjuntos de vectores en un espacio N-dimensional, una máquina de vectores soporte construirá un hiperplano de separación en ese espacio (véase *Figura 31*). Cuanto mayor sea la distancia del hiperplano con los puntos, mejor será el clasificador.

Los vectores de soporte son los puntos que tocan el límite del margen. En el contexto que se está tratado en el presente proyecto de detección de motocicletas, las clases de datos se corresponden a las motocicletas como muestras positivas y a los demás obstáculos como vehículos y peatones como muestras negativas. La SVM busca un hiperplano que separe de forma óptima a los puntos de una clase de la otra, que eventualmente han podido ser previamente proyectados a un espacio de dimensionalidad superior.

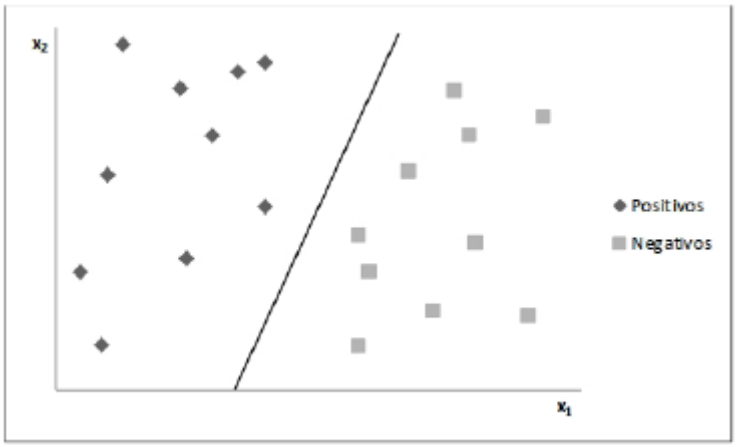


Figura 31: Hiperplano de separación de dos clases.

El concepto de "separación óptima" es donde reside la característica fundamental de las SVM: este tipo de algoritmos buscan el hiperplano que tenga máxima distancia con los puntos que estén más cerca de él mismo. Por este motivo, a veces se les conoce a las SVM como clasificadores de margen máximo. De esta forma, los puntos del vector son etiquetados con una categoría a un lado del hiperplano y los casos que se encuentren en la otra categoría se encontrarán al otro lado.

Para separar linealmente los datos se procede a realizar un cambio de espacio mediante una función que transforme los datos de manera que se puedan separar linealmente. Esta función recibe el nombre de Kernel.

En este caso, los conjuntos son "motocicletas" y "no motocicletas". Para ello, se necesita un entrenamiento previo de la máquina, facilitándole ejemplos de motocicletas o "positivos" y de ejemplos de coches o "negativos". Con todos los ejemplos de entrenamiento, el algoritmo de clasificación SVM elabora una curva M-dimensional que divide ambos conjuntos, obteniendo de esta forma el kernel de la máquina. Las dimensiones del espacio dependen del número de componentes de cada vector a clasificar.

3.3.2. Fundamento matemático.

Las máquinas de vectores soporte presentan un buen rendimiento al generalizar en problemas de clasificación, pese a no incorporar conocimiento específico sobre el dominio. La solución no depende de la estructura del planteamiento del problema.

La idea es construir un clasificador que minimice el error en la separación de los objetos dados y maximice el margen de separación.

Considerando un conjunto de entrenamiento:

$$(\bar{x}_i, z_i) : \bar{x}_i \in \Re^m, i = 1, .., N$$

Figura 32: Conjunto de entrenamiento.

Suponemos que hay un hiperplano que separa los puntos de ambas clases. Los puntos x sobre el hiperplano satisfacen $w^T x + b = 0$ donde el vector w es normal al hiperplano, $\frac{\|b\|}{\|w\|}$ es la distancia perpendicular del hiperplano al origen y $\|w\|$ es la norma euclídea de w.

Si d_+ es la distancia más corta del hiperplano de separación a la muestra positiva más cercana y d_- la distancia más corta del hiperplano de separación a la muestra negativa más cercana, el margen del hiperplano será la suma $d_+ + d_-$. En el caso linealmente separable, el algoritmo buscará simplemente el hiperplano con mayor margen.

Supongamos que todos los datos de entrenamiento satisfacen las siguientes desigualdades:

$$w^T x_i + b > +1 \qquad para \quad z_i = +1$$

$$w^T x_i + b < -1 para \quad z_i = +1$$

$$z_i(w^T x_i + b) - 1 \geq 0 \qquad \forall i$$

Figura 33: Desigualdades de los datos de entrenamiento.

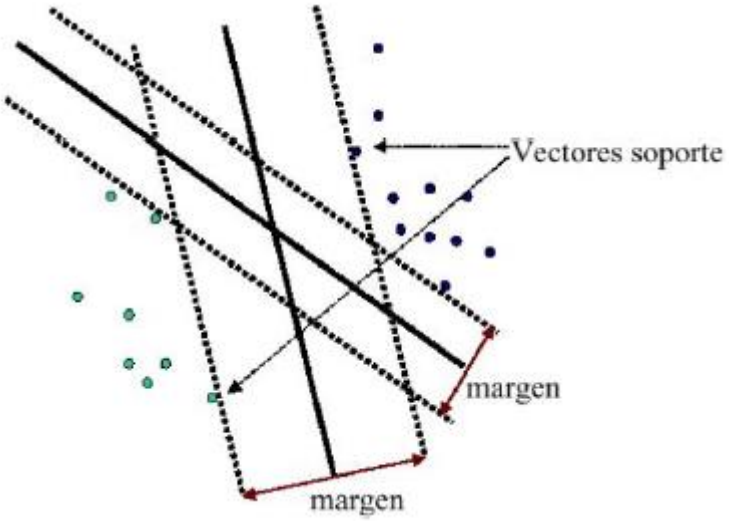


Figura 34: Varias opciones de hiperplano.

Ahora consideramos los puntos para los que se cumple la primera igualdad de la *Figura 33* (que este punto exista es equivalente a elegir una escala adecuada para w y b). Estos puntos están sobre el hiperplano H1: $w^T x_i + b = 1$ con normal w y distancia al origen $\frac{|1-b|}{\|w\|}$.

De la misma forma, para el hiperplano H2 la distancia al origen es $\frac{|-1-b|}{\|w\|}$. Por lo tanto, $d_+ = d = \frac{1}{\|W\|}$ y el margen es $\frac{2}{\|w\|}$. Cabe destacar que H1 y H2 son paralelos por lo que tienen la misma normal y no hay puntos de entrenamiento entre ellos (véase *Figura 35*). Se puede encontrar el par de hiperplanos que dan el máximo margen minimizando la función de coste $\frac{1}{1}\|w\|^2$ con las restricciones de la *Figura 33*.

Ahora pasaremos a una formulación lagrangiana del problema. Hay dos grandes razones para hacer esto:

- Las restricciones de la *Figura 32* se sustituirán por restricciones sobre multiplicadores de Lagrange, que serán más sencillos de utilizar.
- Con esta reformulación, los datos de entrenamiento solo aparecen en forma de productos escalares entre vectores. Esta propiedad es crucial para generalizar el procedimiento al caso no lineal como se verá.

Por lo tanto, introduzcamos N multiplicadores de Lagrange que denotaremos por α1, α2,…, αn uno para cada una de las restricciones de la *Figura 32*.

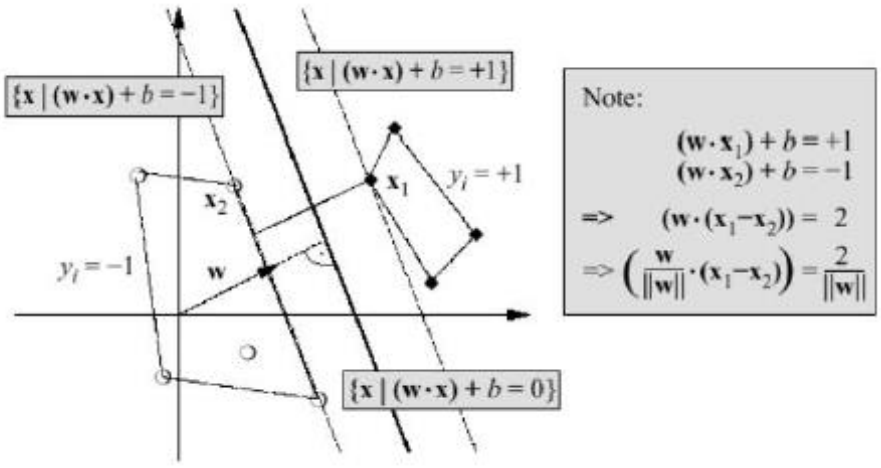


Figura 35: Ilustración de la idea de hiperplano de separación óptimo para el caso de patrones linealmente separables. Los vectores soporte se muestran rodeados por un círculo.

La regla en aplicaciones lagrangianas es que para restricciones de la forma ʒi ≥ 0 las ecuaciones que las definen se multiplican por multiplicadores de Lagrange positivos y se restan de la función objetivo para formar el

lagrangiano. En el caso de restricciones de la forma ʒi = 0, los multiplicadores de Lagrange no tienen restricciones. Lo anterior da el lagrangiano:

$$L_p = \frac{1}{2}||\bar{w}||^2 - \sum_{n=1}^{N} \alpha_i z_i (\bar{w}^T \bar{x}_i + b) + \sum_{n=1}^{N} \alpha_i$$

Figura 36: Lagrangiano.

Ahora debemos minimizar el Lp con respecto a w, b y a la vez exigir que las derivadas de Lp con respecto a todos los αi se anulen, todo sujeto a las restricciones αi ≥ 0. Esto quiere decir que podemos resolver de forma equivalente el siguiente problema dual: maximizar Lp sujeto a la restricción de que el gradiente de Lp con respecto a w y b se anule, y sujeto también a la restricción de que αi ≥ 0. Esta formulación particular del problema se conoce como dual de Wolfe y presenta la probabilidad de que el máximo de Lp con las restricciones ʒ2 ocurre en los mismos valores de w, b y α que el mínimo Lp de con las restricciones ʒ1.

Al requerir que se anule el gradiente de Lp con respecto a w y b, obtenemos las condiciones:

$$\frac{\partial L_p}{\partial \bar{w}} = 0 \Longrightarrow \bar{w} = \sum_{i=1}^{N} \alpha_i z_i \bar{x}_i$$

$$\frac{\partial L_p}{\partial b} = 0 \Longrightarrow \sum_{i=1}^{N} \alpha_i z_i = 0$$

Figura 37: Gradientes de Lp con respecto a w y b nulos.

Ya que estas restricciones aparecen igualdades, podemos sustituirlas en la ecuación:

$$p(x) = \frac{1}{m} \sum_{y_i=1+1|} K(x, x_i)$$

Y obtener:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j z_i z_j \bar{x}_i \bar{x}_j$$

La solución se obtiene minimizando Lp o maximizando LD. Se trata de un problema de programación cuadrática (QP).

Nótese que hay un multiplicador de Lagrange para cada muestra de entrenamiento. Tras obtener una solución, aquellos puntos para los que αi > 0 se denominan vectores soporte y yacen sobre los hiperplanos H1, H2. El resto

de las muestras tienen αi = 0, por ello, el vector w se escribirá como combinación de los vectores soporte.

Con estas máquinas, los vectores soporte son los elementos críticos del conjunto entrenamiento: son los más cercanos a la frontera de decisión y si el resto de puntos no se consideran en un nuevo entrenamiento, el algoritmo encontraría el mismo hiperplano de separación. Los patrones que conformarán el clasificador son los vectores soporte, el resto de patrones de entrenamiento son irrelevantes a los efectos de clasificación.

Podemos observar cómo w está determinado explícitamente por el algoritmo de entrenamiento, pero no es este el caso del umbral b, aunque su obtención es inmediata.

Una vez que hayamos entrenado una máquina de vectores soporte (SVM), para clasificar un patrón de evaluación x basta determinar en qué parte de la frontera de decisión se encuentra y asignarle la etiqueta de la clase correspondiente, es decir, asignamos a x la clase $sgnw^T\ x + b$ donde sgn es la función signo.

4. Descripción General

4.1. Introducción.

El presente proyecto se ha desarrollado con la colaboración del laboratorio de Sistemas Inteligentes de la Universidad Carlos III de Madrid.

A continuación se exponen las distintas herramientas, tanto hardware como software, utilizadas para la elaboración del trabajo.

4.2. Hardware.

En este apartado se describen todos los componentes de hardware que son necesarios para que el algoritmo pueda funcionar correctamente.

4.2.1. Vehículo de pruebas IVVI 2.0.

IVVI 2.0 es el vehículo desarrollado por el Laboratorio de Sistemas Inteligentes de la Universidad Carlos III. Su nombre viene del acrónimo en inglés de Vehículo Inteligente basado en Información Visual y el número indica que se trata del segundo vehículo en el que han instalado los sensores y equipos necesarios para el desarrollo de Sistemas de Ayuda a la Conducción.

Figura 38: Vehículo IVVI 2.0.

El IVVI 2.0 tiene tanto los ordenadores como los sensores o monitores perfectamente integrados dentro del vehículo.

En el vehículo hay instaladas cuatro cámaras en el espectro visible y dos en el infrarrojo. De las cámaras de espectro visible, una está supervisando al conductor para avisar al usuario de los primeros síntomas de sueño. Otras dos cámaras, están dedicadas a la detección de obstáculos para evitar colisiones. Por último, la cámara de espectro visible restante captura en color y detecta las señales de tráfico para comprobar que se está cumpliendo la normativa de

tráfico. Las dos cámaras de infrarrojo se encargan de la detección de obstáculos en condiciones de baja visibilidad.

Figura 39: Ubicación de las cámaras en el IVVI 2.0

La información proveniente de los Sistemas de Ayuda a la Conducción solo se le comunica al conductor cuando el vehículo se encuentra en una situación real de peligro. Se evita así distraer al conductor con información inútil o redundante que solo sirva para distraerle de su tare y suponga un peligro o un estorbo más que una ayuda.

Existen dos formas de hacer llegar esa información: a través de una señal sonora se le avisa del peligro y de la maniobra que debe realizar y a través de una pantalla que se encuentra integrada en el salpicadero se le muestra la información captada y analizada por los sensores.

Figura 40: Pantalla en el IVVI 2.0

Para determinar el estado del vehículo se dispone de una sonda CAN-Bus que obtiene información del funcionamiento del vehículo así como de un sistema GPS-IMU que da la información de la posición y velocidad del vehículo. Tres ordenadores situados en el maletero analizan la información sensorial. Al estar conectados en red pueden intercambiar información de sus respectivos módulos.

Figura 41: Ubicación de los ordenadores en el IVVI 2.0

A continuación se va a proceder a detallar cada uno de los elementos del vehículo de pruebas.

4.2.1.1. Telémetro láser.

El láser que utiliza el vehículo inteligente es un LMS MRS 400001 (*Figura 42*). Funciona con frecuencias de 12.5 Hz, 25 Hz y 50 Hz con un ángulo total de escaneo de 110 ° y con 0.125º, 0.25º ó 0.5º de resolución angular. Su rango de detección es de 50 metros (10% de reflectividad. Se requieren aproximadamente 8 vatios de potencia, que operan mediante un suministro de 12V /24 V.

Figura 42: Telémetro LD-MRS 400001.

La particularidad de este telémetro, que le hace ideal para este tipo de sistemas, es su modo de funcionamiento a esta resolución, que se explicará a continuación.

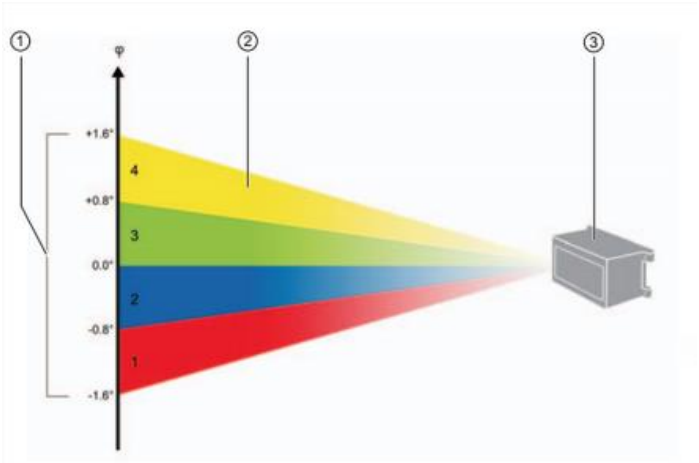


Figura 43: Principio de los planos de exploración del láser LD-MRS.

Estos cuatro planos de exploración se escanean forma entrelazada. Esto significa que la combinación de dos planos se analiza siempre simultáneamente (por ejemplo, primero el amarillo y el plano verde, entonces la azul y el avión rojo). El receptor foto diodo del LD-MRS consta de cuatro receptores independientes dispuestos en una línea. Estos cuatro receptores permiten la aplicación de la tecnología multi-capa. Un receptor se asigna a cada plano, dividiendo así el ángulo de apertura vertical, en cuatro planos de exploración.

Este comportamiento permite identificar los objetos en movimiento debido al patrón que deja a lo largo de las cuatro rotaciones consecutivas. Este patrón será proporcional al movimiento del objeto y su trayectoria a los largo de las cuatro rotaciones.

En el vehículo IVVI 2.0 el escáner láser irá colocado en el parachoques delantero, en un soporte destinado a tal fin.

4.2.1.2. Sistema de visión estéreo.

El IVVI 2.0 contiene una cámara estéreo Bumbleebee XB3 de Point Grey. Esta cámara implementa tecnología de nueva generación que permite minimizar los tiempos de adquisición y mejorar la calidad de datos en 3D.

Figura 44: Cámara Bumbleebee XB3.

Las cámaras estéreo de Point Grey para asegurar la consistencia de la calibración a través de todas las cámaras y eliminar la necesidad de

calibración en el campo calibrado en fábrica tienen una distorsión de la lente de la cámara.

Las imágenes pueden ser rectificadas a cualquier tamaño de la imagen, por lo que es fácil de cambiar la resolución de los resultados de estéreo en función de los requisitos de velocidad y precisión. Calibración de la cámara y la rectificación son clave para conseguir imágenes de disparidad de alta calidad de una cámara estéreo.

Figura 45: Imagen distorsionada vs imagen rectificada.

En el vehículo IVVI 2.0, la cámara estéreo de visión se situará en el interior del vehículo, detrás del parabrisas delantero.

4.2.1.3. Sistema de localización GPS y sensor inercial.

El MTI-G de Xsens es una unidad de medición de pequeño tamaño y bajo peso, excelente para el control y la navegación en sistemas tripulados y otros objetos. El MTI-G es un GPS asistido basado en la unidad de medida inercial (IMU) y un sensor de presión estática. Ofrece un rendimiento sin precedentes por su tamaño, peso, costo y de baja complejidad en su uso. El MTI-G supera los desafíos típicos de la IMU y AHRS que se utilizan en, por ejemplo, aplicaciones aeroespaciales y de automoción.

El diseño del MTI-G es flexible, proporcionando una amplia gama de modos de salida y la configuración avanzada de escenarios de uso específico, la optimización de la fusión de sensores de rutina algoritmo para distintas aplicaciones.

Figura 46: Sistema de localización GPS e inercial MTI-G Xsens

El MTI-G tiene una actitud a bordo y el sistema de partida de referencia (AHRS) y el procesador de navegación. Este procesador de baja potencia de señal digital se ejecuta en tiempo real de sensores Xsens con un algoritmo de fusión, siempre sin desplazamientos, con GPS, y orientación con datos 3D. Además, el MTI-G proporciona una velocidad de datos a un ritmo de actualización más alto que con un típico receptor GPS.

- Los aspectos más destacados son:
  - ✓ Cálculo en tiempo real con GPS de datos de posición y velocidad.
  - ✓ La integración del GPS supera los desafíos típicos de IMU.
  - ✓ AHRS integrado, GPS y sensor de presión estática.
  - ✓ Sensor DSP de abordo funcionando con un algoritmo de fusión.
  - ✓ Frecuencia de actualización alta (120 Hz).
  - ✓ Calibración individual para temperatura, desalineación en 3D y sensor de sensibilidad cruzada.
  - ✓ Salida UTC referenciada.
- Facilidad de uso.

El MTI-G es un sensor que puede ser utilizado en una amplia gama de aplicaciones. Debido a los requisitos específicos para todas estas aplicaciones, el MTI-G utiliza diferentes configuraciones de filtros y restricciones, aplicadas en los escenarios. Entre otros, hay escenarios para el uso en aplicaciones de automoción y aeroespacial.

- Salida.
  - ✓ Orientación 3D (360 °).
  - ✓ Posición y velocidad 3D (con la ayuda de sensores inerciales).
  - ✓ Aceleración 3D, tasa de giro 3D, campo magnético 3D.

4.2.1.4. Visualización.

El vehículo de pruebas IVVI 2.0 está dotado de una pantalla modelo Xenarc 705YV de 7 pulgadas, la cual se utiliza para la

visualización de los resultados, la interacción con los ordenadores a bordo y para mostrar la señal visual de alerta.

Esta pantalla está situada en el salpicadero del vehículo de pruebas.

Figura 47: Pantalla Xernac 705 YV de 7".

4.2.1.5. Alimentación.

El sistema de alimentación está compuesto por una batería de automóvil de 12 V, situada debajo del asiento del conductor y una SAI que realiza la conversión 12V DAC 220 V y mantiene seguros los equipos de bajadas de voltaje o pérdidas de corriente producidas por el fallo de la batería. Los sistemas instalados consumen:

- Consumo del láser: 20 W.
- Consumo de la cámara estéreo: 2.5 W
- Consumo de los ordenadores: 3*90 W = 270 W.
- Consumo de la pantalla: 10 W.

4.3. Software.

En este apartado se describirá el software necesario para la realización del algoritmo de detección de obstáculos para el vehículo inteligente de la Universidad Carlos III de Madrid.

Este proyecto se ha realizado mediante dos software principalmente. Por un lado se ha usado el entorno de desarrollo de Qt Creator en el sistema operativo Ubuntu Linux y el procesamiento de imágenes se ha hecho mediante OpenCV. También se han usado conocimientos de base de datos SQLite y las librerías boost.

4.3.1. Qt Creator.

Qt Creator es un excelente IDE multiplataformas para el desarrollo de aplicaciones en C++ de manera sencilla y rápida. Como su nombre lo indica, está basado en la librería Qt y cuenta con las siguientes características principales:

- Editor avanzado para C++.
- Diseñador de formularios (GUI) integrado.
- Herramientas para la administración y construcción de proyectos.
- Completado automático.
- Depurador visual.

Qt Creator utiliza el compilador de C ++ de la colección de compiladores de GNU con Linux y FreeBSD.

Las características del editor incluyen resaltado de sintaxis y autocompletado, pero no pestañas (aunque los plug-ins están disponibles). Quizás la característica más fuerte es la posibilidad de crear tanto aplicaciones de escritorio como para móviles. Su punto más débil es que el consumo de memoria es un tanto alto.

4.3.2. Librerías OpenCV.

OpenCV es una biblioteca de software desarrollada en 1999 por la compañía Intel para el procesado de imágenes y el desarrollo de la visión por computador. Fue publicada bajo licencia Berkeley Software Distribution (BSD) y es multiplataforma, de forma que se puede utilizar en aplicaciones para GNU/Linux, MacOS X o Windows.

OpenCV se ha desarrollado siguiendo el modelo del software libre, pero está coordinado por el Willow Garage, un grupo de desarrolladores y expertos en robótica.

Pretende proporcionar un entorno de desarrollo fácil de utilizar y altamente eficiente. Esto se ha logrado, realizando su programación en código C y C++ optimizados, aprovechando además las capacidades que proveen los procesadores multi-núcleo. OpenCV además puede utilizar el sistema de primitivas de rendimiento integradas de Intel, que es un conjunto de rutinas de bajo nivel específicas para procesadores Intel.

Las librerías OpenCV disponen de más de 500 funciones que abarcan una gran gama de áreas en el procesamiento de imágenes y se pueden clasificar en los siguientes grupos:

- Procesamiento General de Imágenes.
- Segmentación.
- Transformación.

- Máquina de aprendizaje: Detección y reconocimiento de objetos.
- Calibración de cámaras estéreo o cámaras 3D.
- Árboles de imagen y estructuras de datos.
- Tracking.

Por otra parte, OpenCV incluye cuatro componentes diferentes:

- CxCore: Contiene las estructuras de datos básicas, el soporte XML y las funciones de dibujo.
- CvReference: Incluye todas las funciones para el procesamiento y análisis de imágenes, captura de movimiento, reconocimiento de patrones, calibración de cámaras y análisis de estructura de imágenes.
- CvAux: Contiene funciones para correspondencia estéreo, cambio de forma del punto de vista de las cámaras, seguimiento 3D en estéreo, funciones para el reconocimiento de objetos PCA y modelos de ocultos de Markov embebidos.
- HighGui: Permite crear y abrir ventanas para mostrar las imágenes, leer y escribir gráficos, tanto de imágenes como de vídeo, y manejar de forma simple el ratón, el puntero y otros elementos del teclado.
- Machine Learning: Se emplea para convertir los datos en información mediante la extracción de reglas o patrones de los datos.
- Cv: Contiene las funciones necesarias para las operaciones de procesamiento de imágenes.

Las funciones correspondientes a estas librerías se distinguen porque son nombradas como “cv” más el nombre de la función, mientras que si se trata de tipos de datos se le antepone “Cv” al nombre del tipo de dato.

4.3.3. Base de datos SQLite.

SQLite es un sistema de gestión de bases de datos creado por D. Richard Hip en el año 2000 que se diferencia de bases de datos convencionales como MySQL u Oracle en que esta lee y escribe archivos binarios independientes.

SQlite está escrita en lenguaje C y es de dominio público, por lo que puede ser utilizada en cualquier tipo de proyectos, sean libres o comerciales. Además tiene características muy interesantes:

- Dependiendo de la plataforma, toda la biblioteca se maneja desde un único archivo (shell de comandos) de apenas 500KB.
- Se utiliza el mismo lenguaje de consulta SQL, por lo que de entrada es fácil usarlo.

- Es multiplataforma, incluso hay una gran cantidad de documentación para conectarlo con lenguajes como Java, PHP, Python, .NET, entre otros.
- Las bases de datos se guardan en un fichero con extensión ".db".

Dada su presentación, puede funcionar en dispositivos y máquinas con características de hardware limitadas tales como tabletas o dispositivos móviles.

4.3.4. Librerías Boost C++.

Boost es un conjunto de bibliotecas para el lenguaje de programación C ++ que proporcionan apoyo a las tareas y estructuras tales como álgebra lineal, la generación de números pseudoaleatorios, multilecturas, procesamiento de imágenes, expresiones regulares, y las pruebas unitarias. Además contiene más de ochenta bibliotecas individuales.

Boost se utiliza tan ampliamente debido a que:

- Es de código abierto.
- Proporciona una amplia gama de funcionalidad de plataforma agnóstica que STL pierde.
- Es un complemento a STL en vez de un reemplazo.
- Muchos de los desarrolladores Boost están en la comisión estándar de C++. De hecho, muchas partes de Boost están incluidos en la biblioteca estándar de la siguiente C++.
- Está muy bien documentado.
- Su licencia permite la inclusión en proyectos de código abierto y de código cerrado.
- Por lo general sus características no dependen unas de otras por lo que se puede vincular sólo las partes que se requieren.

5. Desarrollo del algoritmo

En este capítulo, se explican los procedimientos necesarios para la realización del algoritmo, desde el planteamiento inicial hasta la solución final.

En primer lugar, se habla del etiquetador de imágenes, un programa que nos sirve para la extracción de un conjunto de entrenamiento fiable. Posteriormente se desarrollará el entrenamiento del clasificador.

5.1. Etiquetador de imágenes.

La aplicación "Etiquetador.pro" nos servirá para generar una base de datos positivos y negativos de motocicletas y coches con los que poder entrenar el clasificador.

Inicialmente, se dispone de varias secuencias de imágenes captadas por la cámara situada en el interior del vehículo, que muestran la carretera por la que se circula en tiempo real. Estas imágenes se obtienen con unas dimensiones de 640 píxeles de ancho por 480 píxeles de alto.

Una vez que se han obtenido las diferentes secuencias, las debemos ordenar por vistas. Tendremos un total de 8 vistas: frontal, trasera, frontal-derecha, frontal-izquierda, trasera-izquierda, trasera-derecha, lateral izquierda y lateral derecha. Esto se hace para tener una base de datos de imágenes con distintos tipos de vistas y tener un conjunto de entrenamiento más completo y organizado.

Figura 48: Vistas del obstáculo a detectar.

Una vez ordenado todo el conjunto de entrenamiento, el programa "Etiquetador.pro" carga todas las imágenes que se encuentran dentro de la carpeta que se haya seleccionado. Para ello se han utilizado las librerías boost y

las librerías OpenCV: Primero se crea la variable con el “path” en el que se encuentra el conjunto de entrenamiento. Después se va iterando en ese directorio y se van cargando todos los archivos que tengan como extensión “.png”, “.jpg” o “.jpeg”.

```
fs::path directorio("/home/alberto/Escritorio/
                            ConjuntoEntrenamiento/back");
```

Figura 49: Path para cargar las imágenes

```
 for( fs::directory_iterator it( directorio ); it != end_it;
it++ )
        {
    if( fs::is_regular_file( it->status())&& fs::extension(it-
>path())       == ".png" || ".jpg" || ".jpeg")
          {
              std::cout << it->path().filename() << std::endl;
               image = cvLoadImage( it->path().string().data() );
…
}
```

Figura 50: Bucle para cargar imagen a imagen

Posteriormente se debe identificar la región de interés, es decir, crear nuestra imagen ROI que será el obstáculo a detectar, en este caso, las motocicletas como parte positiva del entrenamiento y los coches como parte negativa del entrenamiento.

Dibujar la imagen ROI es uno de los primeros pasos en casi todas las operaciones de cuantificación de imágenes. Es necesario delinear el obstáculo porque solamente nos interesa la información que éste nos proporciona y no su entorno.

Para obtener la imagen ROI se necesita un dispositivo como un joystick, un trackball, o un mouse. También existen métodos automáticos o semi-automáticos. En el programa diseñado para etiquetar las imágenes se realiza mediante el mouse. Para facilitar este proceso se ha creado una especie de cruz en el puntero del mouse que facilita el recorte de la imagen.

Figura 51: Mouse con cruz en el puntero.

Figura 52: Imagen ROI

Para la obtención de la región de interés mediante el mouse, se creó una función en la que dependiendo del evento del mouse, empieza a dibujar el rectángulo o se fija el rectángulo. En este caso al tener el botón izquierdo pulsado, se empieza a dibujar el rectángulo, y cuando sueltas el botón izquierdo termina de dibujar el rectángulo. Además se ha realizado una función para dibujar el rectángulo en la imagen del entrenamiento.

```
void draw_box( cv::Mat img, CvRect rect ){

 if (rect.width <= 0) {rect.x=rect.x+rect.width; rect.width *= -1;}
 if (rect.height <= 0) {rect.y=rect.y+rect.height; rect.height *= -1;}
 cv::rectangle( img, cvPoint(box.x, box.y),
                     cvPoint(box.x+box.width,box.y+box.height),
                     cvScalar(0xff,0x00,0x00) );
}
```

Figura 53: Función para dibujar el rectángulo.

```
void my_mouse_callback( int event, int x, int y, int flags, void*
param )
{

    switch( event ){
        case CV_EVENT_MOUSEMOVE:
            if( drawing_box ){
                box.width = x-box.x;
                box.height = y-box.y;
                temp=copiaimagen.clone();

            }else{
                copiaRecuadro=temp.clone();

        cv::line(copiaRecuadro,cv::Point(0,y),cv::Point(copiaRecuadro.c
        ols,y),cvScalar(0xff,0x00,0x00),1,8,0);

        cv::line(copiaRecuadro,cv::Point(x,0),cv::Point(x,copiaRecuadro
        .rows),cvScalar(0xff,0x00,0x00),1,8,0);

        cv::imshow(name,copiaRecuadro);

            }
            break;
        case CV_EVENT_LBUTTONDOWN:
            drawing_box = true;
            box = cvRect( x, y, 0, 0 );
            break;

        case CV_EVENT_LBUTTONUP:
            drawing_box = false;
            if( box.width < 0 ){
                box.x += box.width;
                box.width *= -1;
            }
            if( box.height < 0 ){
                box.y += box.height;
                box.height *= -1;
            }
            draw_box( temp, box );
            save_box=true;
            break;
    }
}
```

Figura 54: Función de eventos del mouse.

A la hora de recortar la imagen se deben cumplir un par de requisitos:

- En primer lugar el obstáculo debe estar completo ya que sino no están completos no sirven para el entrenamiento.

- Por otra parte, el recorte se debe hacer lo más ajustado al obstáculo para evitar que el clasificar aprenda el fondo. Si tenemos muchas imágenes de una moto quieta y las recortamos con el mismo fondo, el clasificador va a aprender también ese fondo.

A continuación se muestra un ejemplo de cómo se deben hacer los recortes de las imágenes.

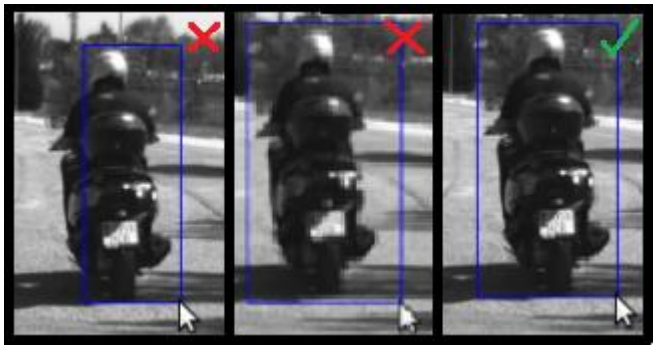


Figura 55: Ejemplo de recorte de la imagen ROI.

Una vez seleccionada la región de interés mediante el mouse, el recorte se guardará automáticamente en una carpeta llamada “recortado”.

Con el fin de aumentar el conjunto de entrenamiento, se crearán las imágenes volteadas respecto del eje vertical, tanto de la imagen original como de la imagen recortada.

El guardado automático de estas 4 imágenes se ha realizado también con las librerías de boost y las librerías de OpenCV. Para ello se crea el “path” correspondiente y se llama a la función de OpenCV “imwrite()”. A esta función se le debe pasar la dirección en la cual se va a guardar la imagen y la imagen que se quiere guardar.

```
nombreOriginal= fs::unique_path(directorio/(it >path().stem().string()
+ ".png"));

nombreUnico= fs::unique_path(recortado/(it->path().stem().string() +
"-%%%%A.png"));

nombreInvertida= fs::unique_path(recortado/
(it >path().stem().string() + "-%%%%B.png"));

nombreOriginalFlipada= fs::unique_path(flipado/
(it->path().stem().string() + "-%%flip.png"));
```

Figura 56: Nombres de las imágenes Original, Recortada, Flipada del recorte, y Flipada de la original.

```
cv::imwrite(nombreUnico.string().data(),image_roi);
```

Figura 57: Función imwrite.

Para generarte la imagen volteada respecto al eje vertical, se ha usado la función "flip" de las librerías de OpenCV. Para ello se debe crear una imagen de tipo Mat en este caso, y pasarle como primer parámetro la imagen original, la imagen creada para hacer el flipado y el tipo de flipado (1 o 0 dependiendo de si se desea un flipado vertical u horizontal respectivamente).

```
cv::Mat image_flip;
cv::flip(image,image_flip,1);
```

Figura 58: Crear variable Mat y función flip.

Por último, se ha diseñado una base de datos en la que se guardarán ciertas características de la imagen ROI. La base de datos se ha creado con SQLiteEl primer paso es crear la base de datos, después se abre la base de datos y se crea una tabla. La misma base de datos puede tener varias tablas. Una vez creada la tabla se pueden insertar los datos.

Para crear una base de datos con SQLite, primero se debe realizar una conexión a la base de datos mediante la clase estática QSqlDatabase. A esta clase se le debe especificar el tipo de base de datos a la que se va a acceder y un nombre de conexión. Una conexión se conoce por su propio nombre y no por el nombre de la base de datos a la que está conectada. Se pueden realizar múltiples conexiones a una base de datos. Si no se le pasa el nombre de la conexión en el argumento se supone la conexión predeterminada. Esto se hace con la función addDatabase de la clase QSqlDatabase.

Una vez creada la base de datos se le debe de dar un nombre. Para ello, se emplea la función setDatabaseName propia de la clase QSqlDatabase. Para que tenga efecto, el nombre de la base de datos se debe establecer antes de abrir la conexión. Esta función no tiene valor por defecto.

```
QString nombre;
nombre.append("BD_Moto_Bar_Sec8_Backright.sqlite");

db = QSqlDatabase::addDatabase("QSQLITE");
db.setDatabaseName(nombre);
```

Figura 59: Nombre y tipo de conexión de la base de datos.

Posteriormente, se debe abrir la conexión a la base de datos. Para ello se utiliza la función "open()".

```
if( db.open()){
      qDebug()<<"Se ha conectado a la base de datos correctamente";
  }
  else {
      qDebug()<<"ERROR! No se ha conectado a la base de datos";
  }
```

Figura 60: Abrir la conexión a la base de datos

Más tarde, como ya se ha comentado, creamos la tabla. Para ello se utiliza la clase QSqlQuery. Esta clase permite ejecutar y manipular sentencias SQL. La clase QSqlQuery encapsula la funcionalidad involucrada en la creación, navegación y la recuperación de los datos de las consultas SQL que se ejecuta en una clase QSqlDatabase. La función "prepare()" de la clase QSqlQuery prepara la consulta SQL para su ejecución. Devuelve el valor True si la consulta se ha realizado con éxito o False si no se ha realizado correctamente. Posteriormente se le añade la sentencia que se quiera ejecutar para la base de datos, en nuestro caso: Create Table. La sintaxis de la sentencia Create Table es la siguiente:

```
CREATE TABLE table_name
(
column_name1 data_type(size),
column_name2 data_type(size),
column_name3 data_type(size),
....
);
```

Figura 61: Sintaxis de la sentencia Create Table.

En el programa “Etiquetador.pro” se ha creado una tabla denominada “Detecciones”. Las columnas de la tabla son las siguientes:

- Nombre de la imagen Original.
- Nombre de la imagen ROI.
- Tipo de obstáculo.
- Vista del obstáculo.
- Ancho del rectángulo.
- Largo del rectángulo.
- Coordenada X de la esquina superior izquierda del recuadro.
- Coordenada Y de la esquina superior izquierda del recuadro.
- Dirección de la imagen original.
- Dirección de la imagen ROI.

```
void crearTablaDetecciones(){
    QSqlQuery query;
    query.prepare("create table detecciones("
                  "original varchar(100),"
                  "crop varchar(100),"
                  "obstacle varchar(100),"
                  "view varchar(100),"
                  "width int,"
                  "height int,"
                  "xSup int,"
                  "ySup int,"
                  "originalDir varchar(100),"
                  "cropDir varchar(100)"
                  ");");

    if(query.exec()){

        qDebug()<<"La tabla DETECCIONES  se ha creado correctamente.";
        qDebug()<<"";

    }else{
        qDebug()<<"La tabla DETECCIONES ya existe o no se ha creado
                  correctamente";
    }
}
```

Figura 62: Crear tabla base de datos SQLite.

Finalmente, queda completar la tabla cada vez que se realiza un recorte. Como cada vez que se obtiene la imagen ROI obtenemos dos imágenes, el propio recorte y el flipado, se insertarán dos filas en la base de datos. Para insertar datos en una base de datos SQL, se emplean las funciones “prepare” (explicada anteriormente) y “bindValue” de la clase QSqlQuery. La función “bindValue” inserta el valor en la posición indicada. En nuestro caso las posiciones representan cada una de las columnas creadas y vienen definidas por las letras del alfabeto. Así por ejemplo para insertar el nombre de la imagen original, se le

indica que tiene que ir en la posición ":a" la cual debe ir a la columna "original" (véase *Figura 63*).

```
void insertarDeteccionNormal()
{
    QSqlQuery insertar;

    insertar.prepare("INSERT INTO
        detecciones(original,crop,obstacle,view,width,
        height,xSup,ySup,originalDir,cropDir)"
       "VALUES(:a,:b,:c,:d,:f,:g,:h,:i,:j,:k)");

    insertar.bindValue(":a",nombreOriginal.filename().string().data());
    insertar.bindValue(":b",nombreUnico.filename().string().data());
    insertar.bindValue(":c", "MOTO");
     //Cambiar interseccion
    insertar.bindValue(":d","BACKRIGHT");
     //*****************************//
    insertar.bindValue(":f",box.width);
    insertar.bindValue(":g",box.height);
    insertar.bindValue(":h",box.x);
    insertar.bindValue(":i",box.y);

    insertar.bindValue(":j",nombreOriginal.parent_path().string().data(
    ));
    insertar.bindValue(":k",nombreUnico.parent_path().string().data());

     //Cambiar interseccion
     insertarDeteccionInvertidaBackLeft();
     //*****************************//
     if(insertar.exec())
      cout<<"Se han insertado datos en DETECCIONES"<<endl;

     else {
         cout <<"NO se han insertado datos en DETECCIONES"<<endl;
         qDebug()<<insertar.lastError()<<endl;
         }
}
```

Figura 63: Insertar detección en base de datos.

Un aspecto importante a la hora de insertar los datos de la imagen flipada es que al reflejar la imagen respecto al eje vertical, la imagen ROI o zona de interés, se desplaza y tiene otra coordenada 'X' mientras que la coordenada 'Y' se mantiene constante. Para obtener la nueva coordenada 'X' y saber dónde se encuentra realmente la zona de interés ha sido necesario realizar un cálculo matemático (véase *Figura 64*).

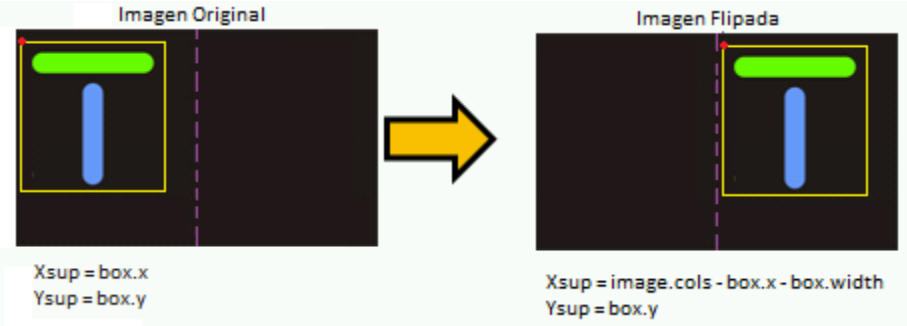
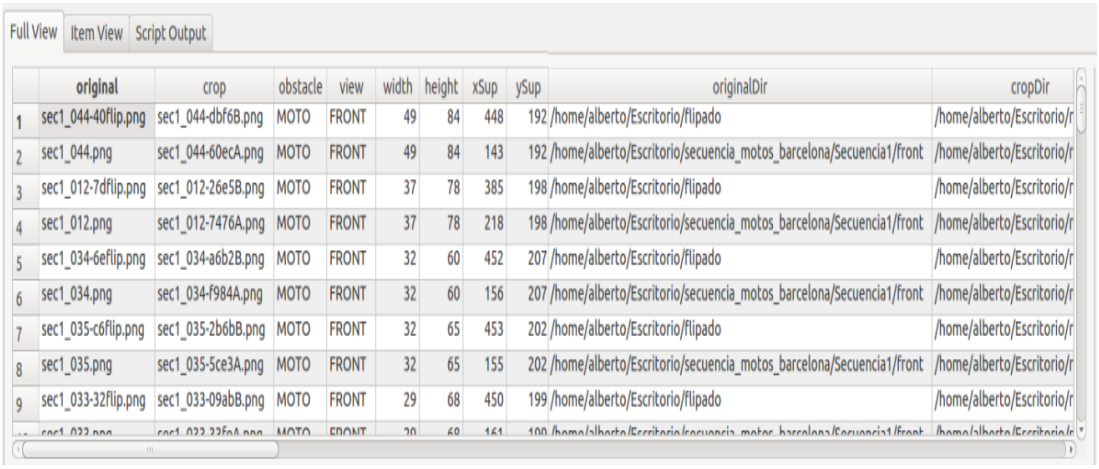


Figura 64: Cálculo coordenada X imagen ROI flipada.

Donde, XSup e Ysup son las coordenadas de la esquina superior de la imagen original, image.cols es el ancho total de la imagen y box.width es el ancho del rectángulo.

Para ver que todo se está guardando correctamente y que la base de datos funciona, se ha descargado el programa Sqliteman, que permite gestionar bases de datos SQLite desde una interfaz muy versátil. En la *Figura 65* se muestra un ejemplo de la base de datos.


Full View | Item View | Script Output

	original	crop	obstacle	view	width	height	xSup	ySup	originalDir	cropDir
1	sec1_044-40flip.png	sec1_044-dbf6B.png	MOTO	FRONT	49	84	448	192	/home/alberto/Escritorio/flipado	/home/alberto/Escritorio/r
2	sec1_044.png	sec1_044-60ecA.png	MOTO	FRONT	49	84	143	192	/home/alberto/Escritorio/secuencia_motos_barcelona/Secuencia1/front	/home/alberto/Escritorio/r
3	sec1_012-7dflip.png	sec1_012-26e5B.png	MOTO	FRONT	37	78	385	198	/home/alberto/Escritorio/flipado	/home/alberto/Escritorio/r
4	sec1_012.png	sec1_012-7476A.png	MOTO	FRONT	37	78	218	198	/home/alberto/Escritorio/secuencia_motos_barcelona/Secuencia1/front	/home/alberto/Escritorio/r
5	sec1_034-6eflip.png	sec1_034-a6b2B.png	MOTO	FRONT	32	60	452	207	/home/alberto/Escritorio/flipado	/home/alberto/Escritorio/r
6	sec1_034.png	sec1_034-f984A.png	MOTO	FRONT	32	60	156	207	/home/alberto/Escritorio/secuencia_motos_barcelona/Secuencia1/front	/home/alberto/Escritorio/r
7	sec1_035-c6flip.png	sec1_035-2b6bB.png	MOTO	FRONT	32	65	453	202	/home/alberto/Escritorio/flipado	/home/alberto/Escritorio/r
8	sec1_035.png	sec1_035-5ce3A.png	MOTO	FRONT	32	65	155	202	/home/alberto/Escritorio/secuencia_motos_barcelona/Secuencia1/front	/home/alberto/Escritorio/r
9	sec1_033-32flip.png	sec1_033-09abB.png	MOTO	FRONT	29	68	450	199	/home/alberto/Escritorio/flipado	/home/alberto/Escritorio/r

Figura 65: Ejemplo de la base de datos.

La ventaja de tener los originales y los recortes anotados en una base de datos es que con un simple script, podemos generar distintos conjuntos de entrenamiento (con más o menos fondo,...) y probar qué tal se comportan.

5.2. Desarrollo del clasificador.

5.2.1. Introducción.

En primer lugar debemos extraer todos aquellos obstáculos que se encuentran en la vía urbana mediante el programa “Etiquetador.pro” explicado anteriormente.

Para tener un buen conjunto de entrenamiento se van a llevar a cabo una serie de descartes y clasificaciones, con el objetivo de que el número de píxeles procesados sea el menor posible, sin que ello afecte a la eficacia de la detección de las motocicletas.

Primero, se van a descartar aquellos obstáculos que se encuentren demasiado lejos del vehículo, ya que no suponen un riesgo de accidente. Por otro lado, no nos interesan aquellas motocicletas o coches que no se vean completos.

Una vez que tenemos todos los obstáculos, se debe entrenar el clasificador. A continuación se detallan los pasos seguidos para la detección.

5.2.2. Obtención de características.

El detector HOG es sencillo de entender. La principal razón de esto es que utiliza una función global para describir a un obstáculo en lugar de un conjunto de características locales, es decir, la totalidad del obstáculo está representada por un único vector de características, a diferencia de muchos vectores que representan partes más pequeñas de la persona.

El detector utiliza una ventana de detección de deslizamiento que se mueve alrededor de la imagen. En cada posición de la ventana del detector, se calcula un descriptor de HOG para la ventana de detección.

Este descriptor se muestra a continuación, a la SVM entrenado, lo que lo clasifica como “motocicleta” o “no motocicleta”.

El objetivo de esta etapa es diferenciar los píxeles que pertenecen al contorno de nuestro obstáculo, en este caso las motocicletas, del resto de píxeles de la imagen. Más tarde, a partir de él se obtiene una estructura que describen puntos y características relevantes de la motocicleta.

Antes de calcular las características de la imagen se deben adecuar las imágenes del entrenamiento:

- ✓ La imagen de la base de datos debe estar en un solo canal de escala de grises para mejor el rendimiento y simplificar los cálculos sucesivos. Si está en tres canales RGB, se debe convertir. Para ello se calcula la media ponderada de los tres canales y el resultado es el valor del canal en escala de grises. A cada píxel se le aplica:

$$RGB[A]toGray: Y \leftarrow 0{,}299 * R + 0{,}587 * G + 0{,}114 * B$$

Figura 66: Conversión de RGB a escala de grises.

El objetivo principal de esta conversión es minimizar el número de píxeles que se tienen que computar. Esta simplificación se puede realizar ya que para hacer el cálculo de bordes, no es relevante los

valores concretos de intensidad de cada canal, sino más bien la variación de intensidad global que se produce.

- ✓ Cambiar el tamaño de la imagen a 64x128.

- ✓ Sobre esta imagen se puede realizar una ecualización del histograma con lo que se maximiza el contraste. Esta técnica aumenta la probabilidad de encontrar el contorno correctamente.

Una vez adaptada la imagen se debe meter las características en un vector de vectores para luego poder entrenar la SVM. Para cada imagen se calcula un vector de características.

Para el cálculo del descriptor HOG, se utilizan celdas de 8 x 8 píxeles dentro de la ventana de detección. Estas celdas se organizan en bloques superpuestos. En la *Figura 67* se puede ver una versión ampliada de una de las imágenes, con una celda de 8 x 8 dibujado en rojo, para hacernos una idea del tamaño de la celda y de la resolución de la imagen en la que se trabaja.

Figura 67: Representación celda 8x8.

Suponemos que queremos calcular el vector gradiente en el píxel en rojo de la *Figura 68*. Ésta es una imagen en escala de grises, por lo que los valores estarán comprendidos entre 0 y 255, donde 0 es negro y 255 es blanco. Los valores de los píxeles a la izquierda y derecha de nuestro píxel están marcados en la imagen: 56 y 94. Si se cogen los valores de derecha a izquierda, el ritmo del cambio en la dirección X es de 38 (94-56 = 38). Se puede calcular el gradiente sustrayendo de izquierda a derecha o al revés, únicamente hay que ser coherente en toda la imagen. Por otro lado, se puede hacer lo mismo para lo píxeles de encima y de debajo, teniendo así el cambio en la dirección Y (93 - 55 = 38).

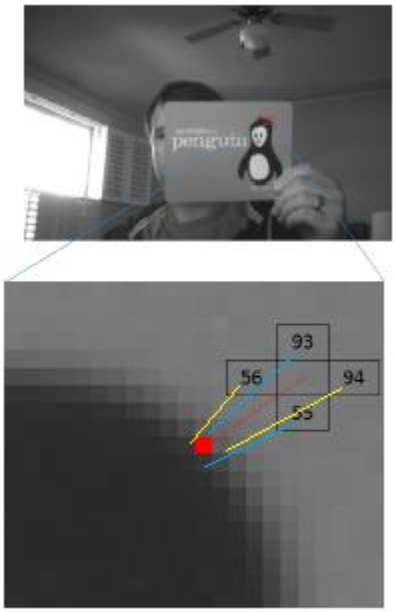


Figura 68: Vector gradiente X Y.

Por lo tanto tenemos el siguiente vector gradiente:

$$\begin{pmatrix} 38 \\ 38 \end{pmatrix}$$

Para obtener la magnitud y dirección de los valores de gradiente en cada uno de los píxeles de la región se utilizan las siguientes expresiones:

$$|G| = \sqrt{I_x^2} + \sqrt{I_y^2}$$

$$\Theta = arctan(\frac{I_x}{I_y})$$

Figura 69: Ecuaciones de módulo y dirección del gradiente.

Cada píxel de la celda tiene un cierto peso en el histograma de orientación, basado en el valor calculado de la magnitud de su gradiente. Cada imagen se representa a través del histograma de cada una de las celdas. Estos histogramas quedan ordenados en un vector según su peso, dando lugar al vector de características de la imagen.

5.2.3. Entrenamiento de la SVM.

Para el entrenamiento de este clasificador se requiere un conjunto de ejemplos positivos y otro negativo. Con ello, como se ha explicado en los fundamentos teóricos, la máquina sitúa estos puntos en un espacio N-dimensional y traza una curva que separa ambos conjuntos. Dependiendo de

la precisión con la que la curva separa ambos conjuntos y de si las muestras tienen características relevantes, obtendremos unos resultados u otros. Para la evaluación del clasificador, necesitaremos un conjunto de muestras de test.

Se ha desarrollado una base de datos específicamente para este proyecto. Consta de diferentes tipos imágenes y archivos de diferentes objetos, con las cuales se enseña y prueba nuestro sistema.

Una vez que se tiene el conjunto de entrenamiento donde tenemos agrupados ejemplos positivos por un lado y ejemplos negativos por otro, se define una metodología para proceder el entrenamiento y el testeo de los diferentes kernels. En este caso, un 70% de las muestras de cada conjunto se utilizará para el entrenamiento, 20% para realizar una realimentación y un 10% para el testeo del clasificador.

A partir de la base de datos y elegidos los dos subconjuntos para cada fase, se procede al entrenamiento del clasificador que dependiendo del método que se utilice en el módulo, se realizará de una forma u otra. Una característica común a todos los métodos es que adquieren la información de entrenamiento siempre de la base de datos. La máquina clasifica estos ejemplos y predice unas etiquetas para cada muestra. De estos resultados se obtienen unos parámetros de comparación que evalúan las prestaciones de la máquina. Finalmente, para cada parámetro se hará la media de cada escenario, obteniéndose al final unos valores promediados y, por tanto, menos dependientes del conjunto seleccionado para el test.

Una vez que se ha obtenido el primer modelo de detección se debe testear con un conjunto de imágenes de prueba para comprobar su eficiencia. Para ello, usamos el 20% de las muestras de realimentación.

Si una imagen con el obstáculo no logró ser detectada, tenemos un falso negativo, y ésta debe ser añadida como imagen positiva a la colección de imágenes de muestra a partir de la cual se calculó el detector. A este proceso se le *llama aprendizaje activo*.

En cambio, si una imagen negativa es detectada, tenemos un falso positivo, y ésta deberá ser añadida a la colección como imagen negativa. Este otro proceso se denomina *bootstrapping*.

Cuando una imagen positiva es detectada y una imagen negativa no lo es, entonces el funcionamiento de nuestro clasificador es correcto, y no se deberá hacer nada con esas imágenes.

Tras realizar esto con todas las imágenes, se volverá a usar el SVM con las nuevas imágenes añadidas, calculando así un nuevo modelo y predecir con el 10% de imágenes destinadas para hacer el test.

5.2.4. Descripción del algoritmo desarrollado.

En este capítulo se explicará cómo se ha desarrollado el algoritmo de la obtención de las características HOG así como del entrenamiento SVM.

Para realizar el cálculo de las características HOG, el entrenamiento del clasificador SVM y la predicción de un conjunto de muestras, se han realizado tres programas.

El primer programa, redimensiona todas las imágenes que están dentro de las carpetas P, N, MejoraPositivas, MejoraNegativas, Testpositivas y Testnegativas. Cuando se obtienen las imágenes ROI son de tamaños variables y para la obtención de las características HOG deben ser múltiplos de 8 x 8 por lo que se debe redimensionar a una imagen de 64x128. La función principal para realizar el redimensionado de una imagen se trata de "*resize*". Los parámetros principales de esta función son los siguientes:

- Imagen de entrada.
- Imagen de salida. Posteriormente se debe guardar la imagen en otro directorio como ya se explicó anteriormente.
- El tamaño de redimensionado.
- Factor de escala a lo largo del eje horizontal.
- Factor de escala a lo largo del eje vertical.
- Interpolación.

```
resize (image, image_red, size,0.5,0.5,INTER_LINEAR);
```

Figura 70: Parámetros de resize.

El segundo programa desarrollado es el más importante. Se trata del cálculo de las características HOG de cada imagen y del entrenamiento del clasificador.

En primer lugar se crea un objeto de la clase HOGDescriptor para crear un detector de motos mediante el histograma de gradientes orientados de OpenCV. Posteriormente debemos decirle a nuestra clase HOG cuál es el tamaño de las imágenes de entrenamiento. Esto se hace llamando a la función "*winSize*" de la clase HOGDescriptor. Esto es un aspecto importante a destacar ya que si no se especifican los tamaños de las imágenes de entrenamiento, el programa no funcionará.

Una vez creada la clase, se abren todas las imágenes una por una y se llama a la función "*calculateFeaturesFromInput*" para el cálculo del vector de gradientes de cada imagen. Los parámetros necesarios son el nombre de la imagen, un vector de características y el hog.

```
calculateFeaturesFromInput(currentImageFile, featureVector, hog);
```

Figura 71: Método "calculateFeaturesFromInput".

Dentro de la función "*calculateFeaturesFromInput*" cabe destacar la función "*hog.compute*". Los argumentos de esta función son:

- La imagen del entrenamiento.
- El vector de salida de descriptores.
- El tamaño del salto de la ventana:
    - La cantidad de solapamiento en la dirección x para la búsqueda de la ventana deslizante.
    - La cantidad de solapamiento en la dirección y para la búsqueda de la ventana deslizante.

  Por lo que si se define como 1x1 la ventana deslizante recorrerá la imagen píxel a píxel.
- Borde alrededor de la imagen. De esta forma el detector puede encontrar cosas cerca de los bordes.
- Vector de localizaciones. Esto sirve para hacer detecciones en lugares concretos. Si se desea una localización completa debe estar vacío.

```
static const Size trainingPadding = Size(0, 0);
static const Size winStride = Size(8, 8);
vector<Point> locations;
hog.compute(imageData, featureVector, winStride, trainingPadding, locations);
```

Figura 72: Método "compute" y sus parámetros.

Para un seguimiento más detallado del entrenamiento, se ha generado un fichero para cada imagen con cada característica hog de cada celda. El número máximo de características hog de cada imagen es 3780 (4 cells x 9 bins x 7 x 15 blocks = 3780). Por otro lado se ha generado un fichero que contiene todas las características HOG de todas las imágenes.

Posteriormente es necesario definir unos parámetros para el entrenamiento del clasificador. Estos parámetros se almacenan en un objeto de la clase CvSVMParams.

El primer parámetro a definir es el tipo de SVM. En este caso se ha elegido el tipo CvSVM::C_SVC que puede ser utilizado para n-clases ($n \geq 2$). Este es el tipo más utilizado. La característica más importante de este tipo es que se ocupa de la separación de los datos de entrenamiento cuando no son linealmente separables. Esta característica no es importante en nuestro caso ya que nuestros datos si son linealmente separables.

El segundo parámetro es el tipo de kernel SVM. La función del kernel es realizar un mapeo con los datos de entrenamiento para mejorar su parecido

con un conjunto linealmente separable de los datos. Este mapeo consiste en aumentar la dimensionalidad de los datos y se realiza de manera eficiente utilizando la función kernel. En nuestro caso hemos utilizado un Kernel Lineal lo que significa que no se realiza ningún mapeo. La selección del kernel apropiado para cada tarea resulta de gran importancia, ya que es éste el que define el espacio de trabajo transformado donde se llevará a cabo el entrenamiento y la clasificación.

Y por último se deben asignar los criterios de terminación del algoritmo. Aquí se especifican el número máximo de iteraciones y un error de tolerancia para permitir que el algoritmo termine en menos pasos, incluso si el hiperplano óptimo no se ha calculado todavía. Este parámetro se define en una estructura cvTermCriteria.

```
// Set up SVM's parameters
   CvSVMParams params;
   params.svm_type    = CvSVM::C_SVC;
   params.kernel_type = CvSVM::LINEAR;
   params.term_crit   = cvTermCriteria(CV_TERMCRIT_ITER, 100, 1e-6);
```

Figura 73: Parámetros SVM.

Una vez definidos los parámetros, entrenamos el clasificador. Para ello, se crea una variable de la clase CvSVM y se llama al método "*train*". Este método tiene los siguientes argumentos:

- TrainingData: Datos de entrenamiento.
- Response: Respuestas de los datos de entrenamiento.
- varIdx: Puede ser nulo si es necesario. Cuando no es nulo, identifica las características de interés.
- SampleIdx: Puede ser nulo si es necesario. Cuando no es nulo, identifica los ejemplos de interés.
- Parámeters: Los parámetros para la SVM explicados anteriormente.

```
SVM.train(trainingDataMat, labelsMat, Mat(), Mat(), params);
```

Figura 74: Método "train" de la clase CvSVM.

Una vez que se ha entrenado, se guarda el clasificador mediante el método "*save*" de la clase CvSVM.

```
SVM.save(descriptorVectorFile.data());
```

Figura 75: Método "save" de la clase CvSVM.

Con esto ya tenemos un clasificador entrenado. Ahora toca predecir las muestras. Para poder predecir las muestras se ha generado otro programa.

El tercer programa se ha diseñado para predecir las muestras de mejora y las muestras de test. En primer lugar se tienen que calcular las características HOG de las imágenes a predecir. Posteriormente antes de poder predecir se debe entrenar el clasificador o cargar un clasificador ya entrenado. En este caso, al tener ya el clasificador entrenado, lo cargaremos mediante el método "*load*".

```
SVM.load("/home/alberto/Escritorio/TFG/descriptorvector.xml",0);
```

Figura 76: Método "load" de la clase CvSVM.

Posteriormente para predecir las muestras hacemos uso del método "*predict*" de la clase CvSVM. Este método tiene como parámetro de entrada la imagen sobre la que se está realizando el testeo.

```
SVM.predict(trainingDataMat.row(i))
```

Figura 77: Parámetro "predict" de la clase CvSVM.

Los datos de la predicción se guardan en un fichero para poder extraer los datos de una forma rápida y sencilla.

Por último, se emplea el predictor en imágenes originales, es decir, de un tamaño distinto al del entrenamiento. Para ello se emplea la función "*detectMultiScale*". Este método pertenece a la clase de los HOGDescriptor. Los parámetros principales de este método son:

- La imagen de tamaño 640x480, es decir, la imagen de captura original, sobre la cual se va a realizar la búsqueda del obstáculo.
- Un vector que son los límites de los objetos detectados.
- Un umbral para la distancia entre las características y la clasificación SVM. Cuanto mayor umbral, más permisivo es el clasificador.
- El salto de la ventana deslizante. Debe ser un múltiplo del bloque deslizante.
- Parámetro Mock para mantener la compatibilidad de interfaz CPU. Debe ser (0,0).
- Coeficiente de la ventana de detección de aumento.
- Coeficiente para regular el umbral de similitud.

Posteriormente se hace un recuadro, mediante la función "*rectangle*" de OpenCV, alrededor del objeto detectado. Para ello se necesita el vector de los límites y la imagen original.

6. Pruebas y resultados

En este capítulo, se mostrarán los resultados de los distintos testeos que se han realizado para comprobar la validez del algoritmo.

En principio se realizarán las pruebas con las imágenes redimensionadas a 64x128 como se ha explicado en la sección anterior. Una vez entrenado veremos qué resultados se obtienen sin realimentación y con realimentación del clasificador.

Posteriormente al tener todos los datos guardados en una base de datos, sabiendo donde se encuentra cada obstáculo con cada imagen original, generamos otro conjunto de entrenamiento de 64x128 pero sin deformar, es decir, añadiéndoles un marco alrededor de la moto. Al hacer esto algunas motos, las que son muy anchas, se cortarán por lo que hay que hacer una selección del nuevo conjunto de entrenamiento. También se debe destacar que el conjunto de negativos del entrenamiento en este caso no serán coches, sino que serán recortes aleatorios de una secuencia en la que no hay ningún coche ni ninguna motocicleta. Se entrenará el clasificador con ese conjunto y se harán las mismas pruebas que con el primer conjunto de muestras, es decir, se realizará una predicción de las muestras de test sin realimentación, y otro con realimentación.

Además estas cuatro pruebas se realizarán con más o menos imágenes de entrenamiento.

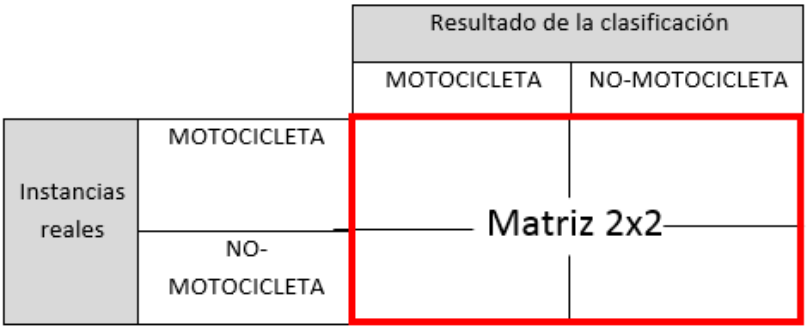
Figura 78: Imagen redimensionada 64x128 vs Imagen con Marco 64x128

Por último se discutirá sobre los resultados de las pruebas de ambos conjuntos del entrenamiento.

6.1. Resultados obtenidos.

Para cada uno de las pruebas se van a calcular las matrices de confusión, los parámetros más comúnmente utilizados para analizar las prestaciones de cada kernel son los siguientes: Accuracy, Precisión, Recall, FP, y MR.

Las matrices de confusión son la herramienta básica que permite visualizar el nivel de confusión de un clasificador.



Donde:

- Las filas son las instancias reales de la clase.
- Las columnas son los resultados de la clasificación para la clase.

En la matriz 2x2 se tiene que:

- Posición [1][1]: Verdaderos Positivos.
- Posición [1][2]: Falsos Negativos.
- Posición [2][1]: Falsos Positivos.
- Posición [2][2]: Verdaderos Negativos.

Números a lo largo de la diagonal principal en la matriz de confusión representan valores que han sido clasificados correctamente, mientras que los que se encuentran fuera de esta diagonal son valores mal clasificados.

La **precisión** nos indica el porcentaje de motos correctamente detectadas que se encuentran dentro de todos los clasificados como motos. Es decir, la calidad de la respuesta del clasificador. Matemáticamente es:

$$Precisión\ (\%) = \frac{TP}{TP + FP}$$

La **exactitud** o **accuracy** es la proporción de detecciones correctas, tanto de verdaderos positivos como de verdaderos negativos, es decir la distancia a la clasificación perfecta:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

El **recall** o **sensibilidad** indica el porcentaje de motos que se han clasificado correctamente, es decir, la eficiencia en la clasificación de todos los elementos que son de la clase:

$$Recall = \frac{TP}{TP + FN}$$

Por el contrario a la sensibilidad, la **Especificidad** es la eficiencia de todos los elementos que no son de la clase:

$$Especifidad = \frac{TN}{TN + FP}$$

El **MR** (**Miss Rate**) nos proporciona el grado de pérdida de candidatos:

$$MR = \frac{FN}{Ntestpos}$$

Donde en todos estos métodos de comparación:

- ✓ **TP**: "**True Positives**" Número de imágenes clasificadas como positivas, siendo positivas.
- ✓ **TN**: "**True Negatives**" Número de imágenes clasificadas como negativas, siendo negativas.
- ✓ **FP**: "**False Positives**" Número de imágenes clasificadas erróneamente como positivas, siendo negativas.
- ✓ **FN**: "**False Negatives**" Número de imágenes clasificadas erróneamente como negativas, siendo positivas.
- ✓ **Ntestpos**: Número de ejemplos de test positivos.

A continuación se muestran los resultados de las distintas pruebas llevadas a cabo en el presente proyecto:

- **<u>Prueba nº 1: Entrenamiento con imágenes de 64 x 128 y predicción de las imágenes de test sin realimentación.</u>**

  Esta prueba consiste, como se ha explicado anteriormente, en entrenar el clasificador con las imágenes de 64x128 pero deformadas. Finalmente se realiza la predicción con las imágenes, redimensionadas a 64x128, destinadas al test.

  Para el entrenamiento de este clasificador se han empleado 5960 imágenes, de las cuales 3260 son positivas y 2700 son negativas. Por otro lado para la predicción se han empleado 596 imágenes (345 positivas y 251 negativas) de una secuencia distinta a la del entrenamiento.

  Los resultados de esta primera prueba han sido los siguientes:

  Matriz de confusión:

	Resultado Clasificación	
	Moto	**No-Moto**
Moto	285	56
No-Moto	64	187

Tabla 1: Matriz de confusión Prueba 1.

Parámetros de comparación:

Precisión (%)	Accuracy (%)	Recall (%)	Especificidad (%)	MR (%)
81.87	79.87	83.77	74.50	22.31

Tabla 2: Parámetros de comparación Prueba 1.

Tasas:

True Positive Rate	False Positive Rate
0.84	0.26

Tabla 3: Tasas TPR y FPR Prueba1.

❖ **Prueba nº 2: Entrenamiento con imágenes de 64 x 128 y predicción de las imágenes de test sin realimentación y con menor número de imágenes de entrenamiento.**

Esta prueba consiste en hacer lo mismo que la anterior pero con menor número de imágenes para entrenar para ver la influencia de este factor.

Para el entrenamiento de este clasificador se han empleado 1700 imágenes, de las cuales 980 son positivas y 720 son negativas. Por otra parte para la predicción se han empleado 386 imágenes (245 positivas y 141 negativas) de una secuencia distinta a la del entrenamiento.

Los resultados de esta primera prueba han sido los siguientes:

Matriz de confusión:

	Resultado Clasificación	
	Moto	**No-Moto**
Moto	201	44
No-Moto	36	105

Tabla 4: Matriz de confusión Prueba 2.

Parámetros de comparación:

Precisión (%)	**Accuracy (%)**	**Recall (%)**	**Especificidad (%)**	**MR (%)**
84.81	79.27	82.04	74.47	31.2

Tabla 5: Parámetros de comparación Prueba 2.

Tasas:

True Positive Rate	**False Positive Rate**
0.82	0.26

Tabla 6: Tasas TPR y FPR Prueba2.

❖ **Prueba nº 3: Entrenamiento con imágenes de 64 x 128 y predicción de las imágenes de test con realimentación.**

En la tercera prueba se entrenará el clasificador con las imágenes de 64x128 deformadas. Posteriormente se realiza la predicción con las imágenes, redimensionadas a 64x128, destinadas a la realimentación.

Se realimentará el clasificador y se volverá a hacer una predicción pero con las imágenes destinadas al testeo.

Para el entrenamiento de este clasificador se han empleado 5960 imágenes, de las cuales 3260 son positivas y 2700 son negativas. La realimentación se hará con un conjunto de muestras de la misma secuencia que del entrenamiento, teniendo así un total de 1592 imágenes, de las cuales 801 son positivas y 791 negativas. Por último para la predicción se han empleado 596 imágenes (345 positivas y 251 negativas) de una secuencia distinta a la del entrenamiento.

Los resultados de esta prueba han sido los siguientes:

Matriz de confusión:

	Resultado Clasificación	
	Moto	**No-Moto**
Moto	786	15
No-Moto	20	771

Tabla 7: Matriz de confusión de la realimentación Prueba 2.

Ahora con estos falsos negativos se realiza un aprendizaje activo y con los falsos positivos se realiza el bootstrapping. Por lo tanto, ahora entrenamos con un conjunto de 3275 y 2720 imágenes positivas y negativas respectivamente. La matriz de confusión de la secuencia de test es:

	Resultado Clasificación	
	Moto	**No-Moto**
Moto	317	28
No-Moto	30	221

Tabla 8: Matriz de confusión Prueba 3.

Parámetros de comparación:

Precisión (%)	Accuracy (%)	Recall (%)	Especificidad (%)	MR (%)
91.35	90.27	91.88	88.05	11.11

Tabla 9: Parámetros de comparación Prueba 3.

Tasas:

True Positive Rate	False Positive Rate
0.92	0.12

Tabla 10: Tasas TPR y FPR Prueba3.

- **Prueba nº 4: Entrenamiento con imágenes de 64 x 128 y predicción de las imágenes de test con realimentación y con menor número de imágenes de entrenamiento.**

En esta prueba se realiza la misma prueba anterior pero con un menor número de imágenes. El conjunto de entrenamiento se realizará con 1700 imágenes en total, 980 son positivas y 720 negativas. La realimentación se realizará con 605 imágenes.

Matriz de confusión:

	Resultado Clasificación	
	Moto	**No-Moto**
Moto	303	20
No-Moto	14	273

Tabla 11: Matriz de confusión de la realimentación Prueba 4.

Ahora con estos falsos negativos se realiza un aprendizaje activo y con los falsos positivos se realiza el bootstrapping. El nuevo entrenamiento tendrá 1734. La matriz de confusión de la secuencia de test es:

	Resultado Clasificación	
	Moto	**No-Moto**
Moto	219	26
No-Moto	23	118

Tabla 12: Matriz de confusión Prueba 4.

Parámetros de comparación:

Precisión (%)	Accuracy (%)	Recall (%)	Especificidad (%)	MR (%)
90.49	87.30	89.39	83.69	18.44

Tabla 13: Parámetros de comparación Prueba 2.

Tasas:

True Positive Rate	False Positive Rate
0.89	0.16

Tabla 14: Tasas TPR y FPR Prueba4.

- **Prueba nº 5: Entrenamiento con imágenes con marco de 64 x 128 y predicción de las imágenes de test sin realimentación.**

En la quinta prueba se entrenará el clasificador con las imágenes de 64x128 con marco, es decir, el la moto no se deformará. Posteriormente se realiza la predicción con las imágenes, destinadas al testeo.

Para el entrenamiento de este clasificador se han empleado 2024 imágenes, de las cuales 1274 son positivas y 750 son negativas. Para la

predicción se han empleado 325 imágenes (175 positivas y 200 negativas) de una secuencia distinta a la del entrenamiento.

Los resultados de esta prueba han sido los siguientes:

Matriz de confusión:

	Resultado Clasificación	
	Moto	**No-Moto**
Moto	133	42
No-Moto	24	176

Tabla 15: Matriz de confusión Prueba 5.

Parámetros de comparación:

Precisión (%)	Accuracy (%)	Recall (%)	Especificidad (%)	MR (%)
84.71	82.40	76.00	88.00	21.00

Tabla 16: Parámetros de comparación Prueba 5.

Tasas:

True Positive Rate	False Positive Rate
0.76	0.12

Tabla 17: Tasas TPR y FPR Prueba5.

- **Prueba nº 6: Entrenamiento con imágenes con marco de 64 x 128 y predicción de las imágenes de test sin realimentación con menor número de imágenes.**

Esta sexta prueba consiste en lo mismo que la prueba 5 pero con menor número de imágenes.

Para el entrenamiento de este clasificador se han empleado 1012 imágenes, de las cuales 637 son positivas y 375 son negativas. Para la

predicción se han empleado 300 imágenes (160 positivas y 140 negativas) de una secuencia distinta a la del entrenamiento.

Los resultados de esta prueba han sido los siguientes:

Matriz de confusión:

	Resultado Clasificación	
	Moto	**No-Moto**
Moto	132	28
No-Moto	26	114

Tabla 18: Matriz de confusión Prueba 6.

Parámetros de comparación:

Precisión (%)	Accuracy (%)	Recall (%)	Especificidad (%)	MR (%)
83.54	82.00	82.50	81.42	20.00

Tabla 19: Parámetros de comparación Prueba 6.

Tasas:

True Positive Rate	False Positive Rate
0.82	0.19

Tabla 20: Tasas TPR y FPR Prueba6.

- **Prueba nº 7: Entrenamiento con imágenes con marco de 64 x 128 y predicción de las imágenes de test con realimentación.**

En esta séptima prueba se entrenará el clasificador con las imágenes de 64x128 con marco como con las dos anteriores. Posteriormente se una realimentación con imágenes de la misma secuencia. Finalmente se realizará la predicción con las imágenes destinadas al testeo.

Para el entrenamiento de este clasificador se han empleado 2024 imágenes, de las cuales 1274 son positivas y 750 son negativas. . La realimentación se hará con un conjunto de muestras de 650 imágenes, de las cuales 400 son positivas y 250 negativas. Para la predicción se han

empleado 325 imágenes (175 positivas y 200 negativas) de una secuencia distinta a la del entrenamiento.

Los resultados de esta prueba han sido los siguientes:

Matriz de confusión:

	Resultado Clasificación	
	Moto	**No-Moto**
Moto	385	15
No-Moto	9	241

Tabla 21: Matriz de confusión de la realimentación Prueba 7.

Ahora, como en la prueba 2 y prueba 4, se realiza un aprendizaje activo y el bootstrapping. El conjunto de entrenamiento en este caso consta de 2048 imágenes. La matriz de confusión de la secuencia de test es:

	Resultado Clasificación	
	Moto	**No-Moto**
Moto	160	12
No-Moto	6	188

Tabla 22: Matriz de confusión Prueba 7.

Parámetros de comparación:

Precisión (%)	Accuracy (%)	Recall (%)	Especificidad (%)	MR (%)
96.38	95.08	93.02	96.91	6.00

Tabla 23: Parámetros de comparación Prueba 7.

Tasas:

True Positive Rate	False Positive Rate
0.93	0.03

Tabla 24: Tasas TPR y FPR Prueba7.

- **<u>Prueba nº 8: Entrenamiento con imágenes con marco de 64 x 128 y predicción de las imágenes de test con realimentación y con menor número de imágenes.</u>**

  En esta última prueba se entrenará el clasificador de la misma forma que la prueba nº 7 pero con menor número de imágenes.

  Para el entrenamiento de este clasificador se han empleado 1012 imágenes, de las cuales 637 son positivas y 375 son negativas. . La realimentación se hará con un conjunto de muestras de 325 imágenes, de las cuales 200 son positivas y 125 negativas. Para la predicción se han empleado 300 imágenes (160 positivas y 140 negativas) de una secuencia distinta a la del entrenamiento.

  Los resultados de esta prueba han sido los siguientes:

  Matriz de confusión:

	Resultado Clasificación	
	Moto	**No-Moto**
Moto	192	8
No-Moto	8	117

Tabla 25: Matriz de confusión de la realimentación Prueba 8.

Ahora, como en la prueba 2, se realiza un aprendizaje activo y el bootstrapping. En este caso tenemos un conjunto de entrenamiento de 1028 imágenes, de las cuales 645 son positivas y 383 negativas. La matriz de confusión de la secuencia de test es:

	Resultado Clasificación	
	Moto	No-Moto
Moto	148	12
No-Moto	14	126

Tabla 26: Matriz de confusión Prueba 8.

Parámetros de comparación:

Precisión (%)	Accuracy (%)	Recall (%)	Especificidad (%)	MR (%)
91.36	91.33	92.5	90.00	8.57

Tabla 27: Parámetros de comparación Prueba 8.

Tasas:

True Positive Rate	False Positive Rate
0.93	0.10

Tabla 28: Tasas TPR y FPR Prueba8.

6.2. Discusión de los resultados.

En este apartado, se van a comparar los resultados de las distintas pruebas llevadas a cabo. La siguiente tabla muestra los valores de precisión, exactitud, sensibilidad, especificidad y MR de las 8 pruebas:

	Precisión (%)	Accuracy (%)	Recall (%)	Especificidad (%)	MR (%)
Prueba 1	81.86	79.87	83.77	74.50	22.31
Prueba 2	84.81	79.27	82.04	74.47	31.21
Prueba 3	91.35	90.27	91.88	88.05	11.15
Prueba 4	90.50	87.31	89.38	83.69	18.43
Prueba 5	84.71	82.40	76.00	88.00	21.00
Prueba 6	83.54	82.00	82.50	81.43	20.00
Prueba 7	96.38	95.08	93.02	96.91	6.00
Prueba 8	91.36	91.33	92.50	90.00	8.57

Tabla 29: Comparativa de las Pruebas.

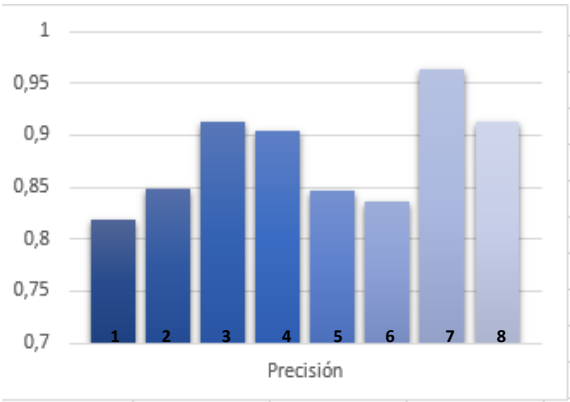


Diagrama según Precisión.

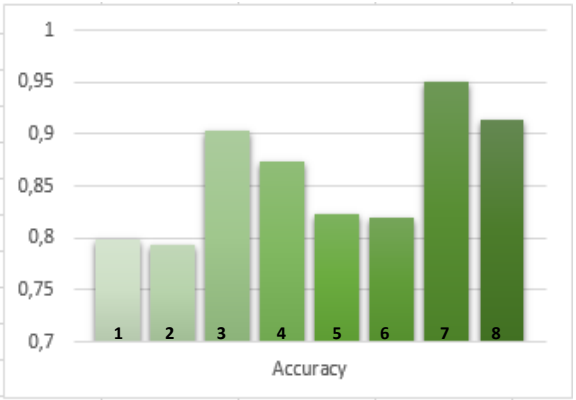


Diagrama según Accuracy.

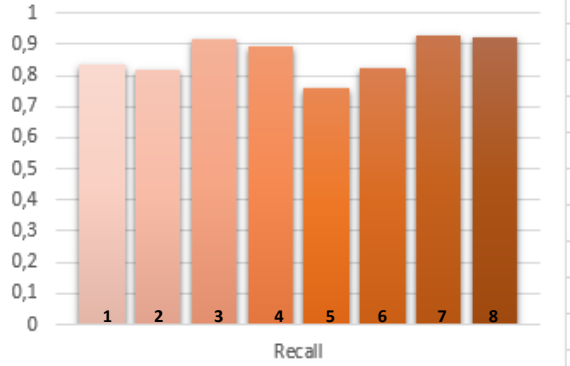


Diagrama según Recall.

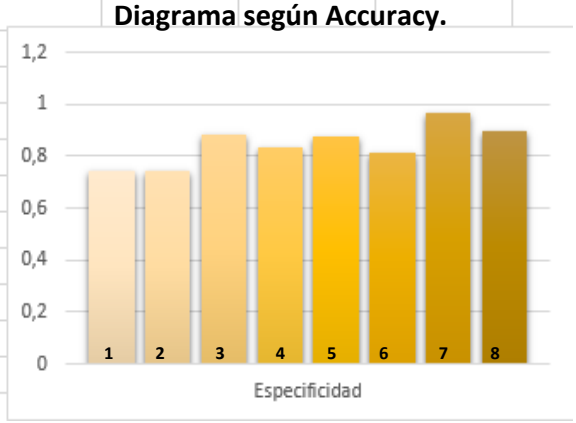


Diagrama según Especificidad.

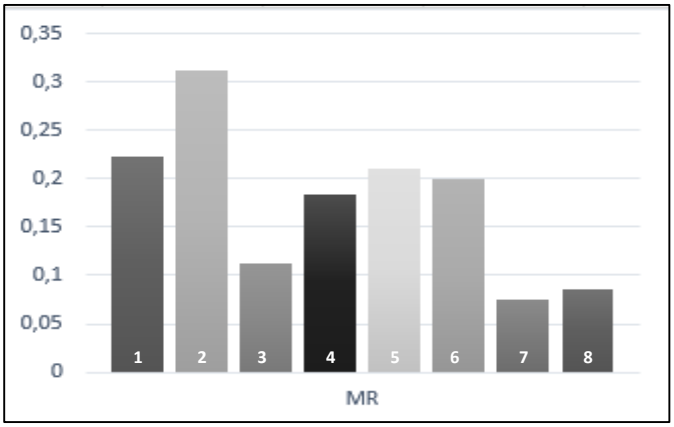


Diagrama según MR (Miss Rate).

Si comparamos cada prueba según el número de imágenes de entrenamiento, se observa que cuanto mayor número de imágenes de entrenamiento, más preciso y exacto es nuestro clasificador. Sin embargo, la sensibilidad y la especificidad no presentan una gran diferencia. Finalmente, en cuanto al MR (Miss Rate) o pérdida de candidatos, cuando se entrena con menos imágenes, se tienen mejores resultados, es decir, tenemos menor pérdida de candidatos.

Comparando los resultados según el método de redimensionamiento de la imagen de entrenamiento, es decir, si la imagen ha sido redimensionada a 64x128, deformándose un poco la imagen, o si se ha añadido un marco alrededor del obstáculo, se observa que todos los casos se obtienen mejores resultados, aunque muy parecidos, de precisión, exactitud, sensibilidad, especificidad y MR.

Por último si comparamos los resultados en función del proceso de predicción, es decir, si se ha realizado realimentación o no, se observa una clara mejoría en todos esos casos en los que si se ha realizado una realimentación. El re-entrenamiento es una característica fundamental del entrenamiento SVM

Por lo tanto, el mejor clasificador es el de la Prueba 7, es decir, un entrenamiento con imágenes con marco de 64x128, re-entrenando el clasificador y con un número de entrenamiento considerable.

7. Líneas futuras

En este capítulo se hablarán de las posibles mejoras que se pueden realizar a algoritmo desarrollado en el proyecto:

- **Combinar el dominio visible con cámaras infrarrojas**. Este tipo de cámaras no dependen en gran medida de la iluminación y ofrecen un menor nivel de ruido, por lo que hace mucho más sencillo la detección de motocicletas. Como desventaja, las cámaras infrarrojas carecen de texturas y colores, dando lugar a pérdida de información. Combinando ambos sistemas obtendremos mejores resultados.

- **Agregar la detección de otros obstáculos**. Un proyecto futuro viable sería la detección de otros obstáculos presentes en la carretera como puede ser los peatones o los propios vehículos. Simplemente habría que modificar la base de datos y agregar este tipo de obstáculos.

- **Ayuda de frenado**. Si en un futuro se le hiciera al vehículo de pruebas su correspondiente modificación tendría que ser capaz de realizar una frenada de emergencia por sí sólo si fuera necesario.

- **Sistema de alerta**. Una posible ampliación del presente proyecto sería en programar un sistema de alerta que se active en caso de que la motocicleta este en situación de un posible accidente. Este sistema de alerta podría ser sonoro o con una vibración en el volante como se produce en algunos de los sistemas ADAS descritos en el estado del arte.

- **Seguimiento de obstáculos.** Una vez detectado la motocicleta sería interesante poder predecir la trayectoria de la misma. Esto nos permitiría saber si la trayectoria de la motocicleta es la misma que la de nuestro vehículo o si no lo es. Para este posible proyecto haría falta recurrir al filtro Kalman. El filtro Kalman es un algoritmo recursivo que puede correr en tiempo real usando únicamente las mediciones de entrada actuales, el estado calculado previamente y su matriz de incertidumbre, no requiere alguna otra información pasada adicional.

- **Detección de obstáculos con láser.** Otro trabajo futuro es el de poder incorporar al proceso de detección el láser del vehículo inteligente de la Universidad Carlos III de Madrid. Gracias a la fusión de estos sensores (cámara estéreo y láser), se obtendrán mejores resultados.

8. Conclusiones

El objetivo principal de este proyecto era el desarrollo de un algoritmo capaz de detectar y localizar obstáculos en entornos urbanos mediante visión.

Se optó desde un primer momento por la combinación de HOG-SVM ya que está comprobado que la técnica de HOG nos extrae las características de la imagen de una forma robusta y que el clasificador SVM desarrollado proporciona resultados rápidos aunque tiene algún falso positivo y falso negativo.

La ventaja de este tipo de máquinas clasificadoras es la posibilidad que permiten de aprendizaje. Esto es, la posibilidad de ser re-entrenadas con todas aquellas muestras que en otras ocasiones ha clasificado mal. Este hecho hace a la máquina más robusta con cada entrenamiento.

Como se ha observado en los resultados, las imágenes redimensionadas a 64x128 proporcionan peores resultados que a las que se les ha añadido un marco, aunque con estas últimas se haya entrenado con muchas menos imágenes ya que se ha tenido que hacer una clasificación al tener varias motocicletas no completas. Esto se debe a que al redimensionar la imagen, el obstáculo se distorsiona, empeorando así los bordes reales de una motocicleta.

Puesto que se ha generado una base de datos bastante amplia y con un gran conjunto de modelos de motocicletas, el algoritmo está dando buenos resultados. La principal ventaja de este proyecto es la generación de nuestro propio conjunto de entrenamiento, lo cual hace sencillo la predicción de motocicletas.

9. Bibliografía

[1] Seguridad Vial, “Menos tráfico, máximo riesgo”. http://asp-es.secure-zone.net/v2/index.jsp?id=5938/10033/21340&lng=es&startPage=10

[2] Conducción segura en el automóvil del futuro, http://www.coches.net/noticias/conduccion-segura-en-el-automovil-futuro

[3] Evolución de los sistemas de seguridad, http://www.mapfre.com/ccm/content/documentos/fundacion/seg-vial/investigacion/evolucion-sistemas-seguridad-2006-2011.pdf

[4] Sistema de Control de Crucero Adaptativo (ACC), http://www.anfac.com/openPublicPdf.action?idDoc=7883

[5] El Golf 2013 Sistemas de asistencia al conductor, http://es.slideshare.net/cerjs/el-golf-2013-sistemas-de-asistencia-al-conductor

[6] Detección de objetos en ángulo muerto http://www.km77.com/glosario/d/deteccion.asp

[7] Esteban José Domínguez y Julián Ferrer, “Circuitos eléctricos auxiliares”. https://books.google.es/books?id=mdv2AwAAQBAJ&pg=PA274&lpg=PA274&dq=las+funciones+de+los+sistemas+ACC+m%C3%A1s+avanzados+son+las+siguientes&source=bl&ots=YU4mRV7LaZ&sig=VTXjPrq2TTLIyvDnSmiA2lWxoEs&hl=es&sa=X&ved=0CCEQ6AEwAGoVChMImPSgjt2RyAIVRdIaCh1oSw5a#v=onepage&q=las%20funciones%20de%20los%20sistemas%20ACC%20m%C3%A1s%20avanzados%20son%20las%20siguientes&f=false

[8] Control de estabilidad, https://es.wikipedia.org/wiki/Control_de_estabilidad

[9] Sistema antibloqueo de ruedas, https://es.wikipedia.org/wiki/Sistema_antibloqueo_de_ruedas

[10] Coches Autónomos accidentes, http://www.cochesinconductor.com/2015/07/podrian-los-coches-autonomos-chocar-entre-si/

[11] Capítulo 2. Generación, Representación y Principios de Procesamiento Digital de Imágenes, http://catarina.udlap.mx/u_dl_a/tales/documentos/lem/jimenez_c_e/capitulo2.pdf

[12] Técnicas de procesamiento de imágenes, http://www.cesfelipesegundo.com/revista/articulos2011/Guerrero,%20J.M.pdf

[13] Sistemas inteligentes de transporte uc3m, http://portal.uc3m.es/portal/page/portal/dpto_ing_sistemas_automatica/investigacion/lab_sist_inteligentes/sis_int_transporte

[14] Vehículo IVVI 2.0. uc3m, http://www.uc3m.es/portal/page/portal/dpto_ing_sistemas_automatica/investigacion/lab_sist_inteligentes/sis_int_transporte/vehiculos/Ivvl20/
[15] MTi-G, xsens, https://www.xsens.com/products/mti-g/
[16] Camara stereo Vision BlumblebeeXB3, http://www.ptgrey.com/bumblebee-xb3-1394b-stereo-vision-camera-systems-2
[17] Primeros pasos en sqlite, http://rafinguer.blogspot.com.es/2009/09/primeros-pasos-en-sqlite.html
[18] OpenCV 2 Computer Vision Application Programming Cookbook.
[19] Procesamiento de imágenes en Medicina Nuclear, http://www.alasbimn.net/comites/tecnologos/material/Procesamiento_de_imagenes.pdf
[20] Qsqldatabase Class, http://doc.qt.io/qt-5/qsqldatabase.html#details
[21] Qsqlquery Class, http://doc.qt.io/qt-4.8/qsqlquery.html#details
[22] Reconocimiento visual de manos, http://eprints.ucm.es/16073/1/Interaccion_personacomputador_basada_en_el_reconocimiento_visual_de_manos.pdf
[23] Reflejo de una imagen, http://www.openrtm.org/openrtm/ja/content/rt%E3%82%B3%E3%83%B3%E3%83%9D%E3%83%BC%E3%83%8D%E3%83%B3%E3%83%88%E4%BD%9C%E6%88%90opencv%E7%B7%A8-rtcb-rc1
[24] Propuesta de descriptor basado en partes para el reconocimiento de expresiones y objetos en secuencias de imágenes, http://www.academia.edu/11062309/Propuesta_de_descriptor_basado_en_partes_para_el_reconocimiento_de_expresiones_y_objetos_en_secuencias_de_ima_genes
[25] W. T. Freeman and M. Roth, "Orientation histograms for hand gesture recognition," in International Workshop on Automatic Face and Gesture Recognition, vol. 12, pp. 296–301, 1995.
[26] D. G. Lowe, "Object recognition from local scale-invariant features," in Computer vision, 1999. The proceedings of the seventh IEEE international conference on, vol. 2, pp. 1150–1157, Ieee, 1999.
[27] J. Malik, S. Belongie, T. Leung, and J. Shi, "Contour and texture analysis for image segmentation," International journal of computer vision, vol. 43, no. 1, pp. 7–27, 2001.
[28] Detección de Personas en secuencias de vídeo en tiempo real, https://riunet.upv.es/bitstream/handle/10251/12624/TesinaJavier_Oliver_Moll.pdf?sequence=1
[29] Joachims, T. (2008). SVM-Light Support Vector Machine. [en línea] disponible en: http://svmlight.joachims.org.
[30] Dalal, N. y Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. Montbonnot, France.

[31] Implementación de un algoritmo para el reconocimiento y análisis de peatones utilizando visión por computador, http://repository.javeriana.edu.co/bitstream/10554/13621/1/CortesSanabriaDanielFelipe2013.pdf
[32] Support Vector Machines, SVM http://bibing.us.es/proyectos/abreproy/11185/fichero/Volumen+1_Detector+Multiusuario+para+DS-CDMA+basado+en+SVM%252F7.+Support+Vector+Machines%252FSupport+Vector+Machines.pdf
[33] HOG Person Detector Tutorial by Chris MCCormick, https://chrisjmccormick.wordpress.com/2013/05/09/hog-person-detector-tutorial/
[34] OpenCV docs "Introduction to SVM", http://docs.opencv.org/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html
[35] OpenCV docs "Support Vector Machine SVM", http://docs.opencv.org/modules/ml/doc/support_vector_machines.html
[36] What does support vector machine (SVM) mean in layman's terms? http://www.quora.com/What-does-support-vector-machine-SVM-mean-in-laymans-terms
[37] Performance Characterization in Computer Vision, http://vase.essex.ac.uk/talks/performance-evaluation.pdf
[38] Verificación de vehículos mediante técnicas de visión artificial, http://arantxa.ii.uam.es/~jms/pfcsteleco/lecturas/20140714GonzaloBallesterosVillareal.pdf
[39] C. J.Watkins and P. Dayan, "Q-learning," Machine learning, vol. 8, no. 3-4, pp. 279–292, 1992.
[40] C. Cortes and V. Vapnik, "Support-vector networks," Machine learning, vol. 20, no. 3, pp. 273–297, 1995.
[41] http://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf